Reconocimiento de Escritura
Lecture 2/5 — Isolated Character Recognition

Daniel Keysers

Jan/Feb-2008

Outline

Introduction

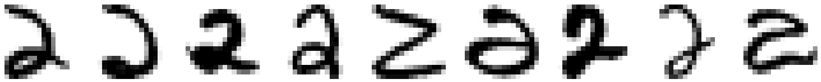
off-line handwriting

- single characters are easily segmented
  - forms with boxes
  - postal codes
  - $\rightarrow$ this lecture
- single characters are difficult to segment
  - continuous text
  - use segmentation hypotheses $\rightarrow$ next lecture
  - use HMM-based approach $\rightarrow$ Alejandro

Image Variability

In handwriting recognition there is significant amount of variability present in the images to be processed.

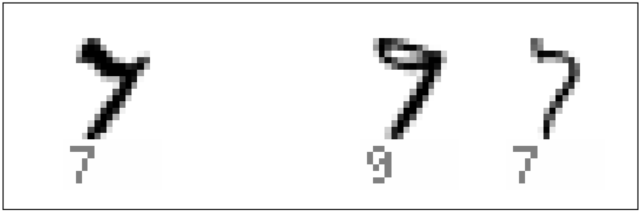
We will discuss several methods to deal with this variability



Typical Data Sets

name	example images	size	#train	#test
USPS		16×16	7 291	2 007
MNIST		28×28	60 000	10 000

Invariance



invariance requirements in classification
$=$ prior knowledge about $p(x|k)$ (informally: $p(x|k) \cong p(t(x,\alpha)|k)$)
$=$ suitable model for $d(x, x_{kn})$ ($d(x, x_{kn}) := \min_{\alpha} d'(x, t(x_{kn}, \alpha))$)

other possibilities:
– feature analysis $\rightarrow$ invariant features
– preprocessing $\rightarrow$ normalization
– references $\rightarrow$ virtual data

Invariance

Different approaches:

- normalize original image:
  eliminate transformation prior to feature extraction
  (e.g. using moments)
- invariant features:
  - histograms (RT invariant)
  - Hu-moments (RST invariant)
  - Fourier-Mellin transform (RST invariant)
  - integral features (RT invariant)
  - ...
- virtual data: add artificially created training data
- appearance-based approach
  (i.e. interpret the image itself as feature vector)
  and allow for transformations during recognition

Virtual Data

Typical drawback of learning classifiers:

- insufficient amount of training data
  $\rightarrow$ create virtual training data

choose 'suitable' transformation t with parameter α,
create virtual data by applying t to the training data

Effects:

- we gain additional training data, leading to more reliable parameter estimation
- local invariance with respect to t

Simple example:

- choose ± 1 pixel shifts
- 9 fold increase in training samples

Extension:
apply this idea to the testing data, too.
(inspired by classifier combination schemes)
$\rightarrow$ Virtual-Test-Sample Method

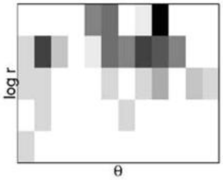
Polynomial Classifiers

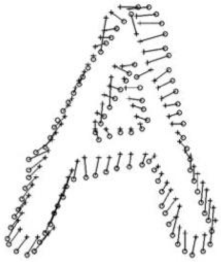
Polynomial Classifiers

- one of the oldest methods for digit classification
- still in use in many postal systems today
- fast and small
- can be used in a hierarchical way
- general framework: function approximation
- training usually simple
- sometimes lower error rates achieved by other methods

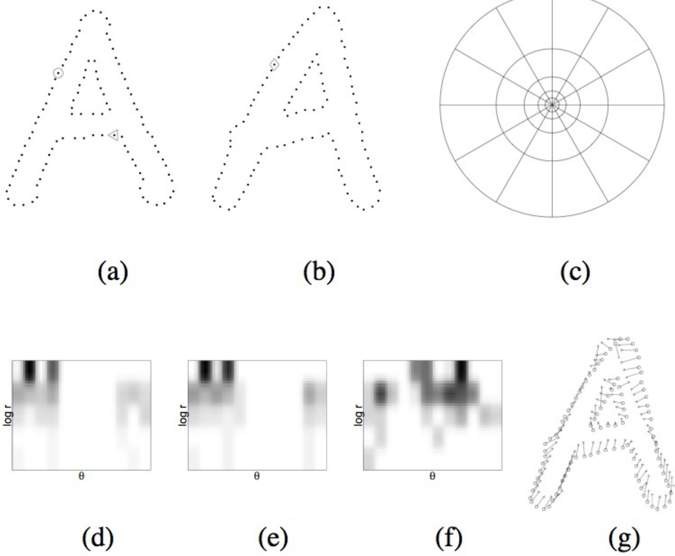
Shape Context Matching (Belongie et al.)

Belongie & Malik$^+$ 2002 (Berkeley)
shape contexts = log-polar histograms of contour points
iterative matching with 2D-splines and the Hungarian algorithm



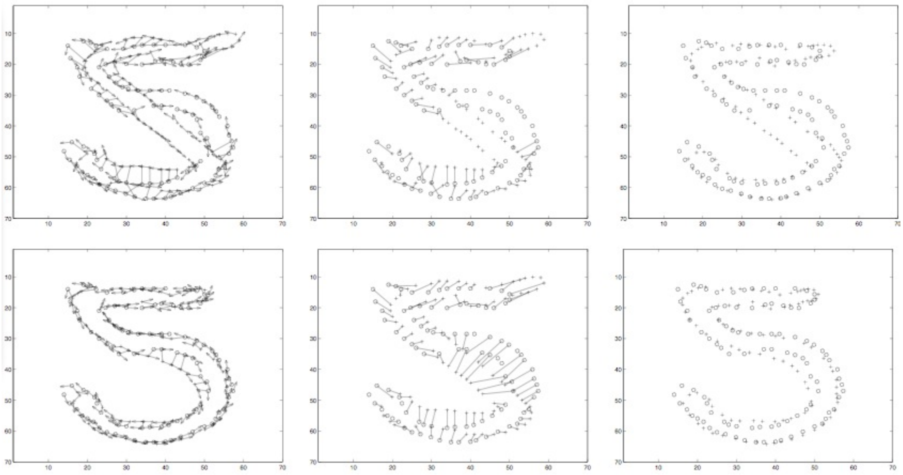


Shape Context Matching



descriptors, cp. SIFT (=Scale Invariant Feature Transform)

Shape Context Matching

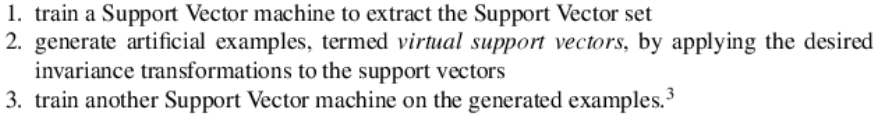


iterative matching, using 2D-spline regularization

Invariant Support Vector Machines (DeCoste et al.)

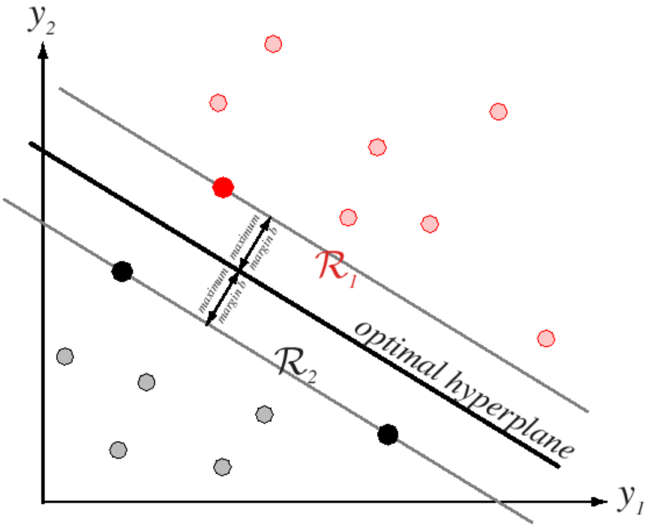
D. DeCoste (CalTech/MSR), B. Schölkopf (MPI):
Training Invariant Support Vector Machines.
Machine Learning, 46, 161190, 2002

use virtual data and kernel jittering in support vector machine



1. train a Support Vector machine to extract the Support Vector set
2. generate artificial examples, termed *virtual support vectors*, by applying the desired invariance transformations to the support vectors
3. train another Support Vector machine on the generated examples.[3]

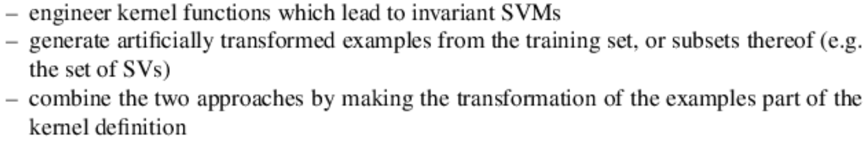
SVM


y_2
y_1
maximum margin b
maximum margin b
$\mathcal{R}_1$
$\mathcal{R}_2$
optimal hyperplane

Invariance

“Practical experience has shown that in order to obtain the best possible performance, prior knowledge about invariances of a classification problem at hand ought to be incorporated into the training procedure.”

in SVMs:

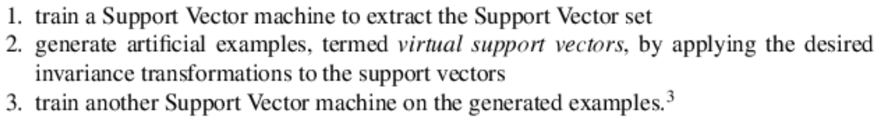


– engineer kernel functions which lead to invariant SVMs
– generate artificially transformed examples from the training set, or subsets thereof (e.g. the set of SVs)
– combine the two approaches by making the transformation of the examples part of the kernel definition

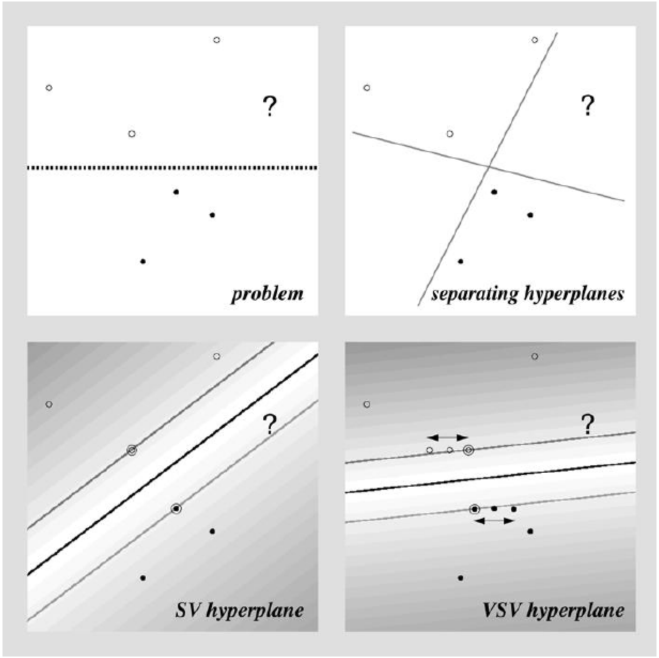
Method 1: Virtual SVs

SV set contains all information necessary to solve a given classification task.

It might be sufficient to generate virtual examples from the Support Vectors only.



1. train a Support Vector machine to extract the Support Vector set
2. generate artificial examples, termed *virtual support vectors*, by applying the desired invariance transformations to the support vectors
3. train another Support Vector machine on the generated examples.[3]


?
problem
?
separating hyperplanes
?
SV hyperplane
?
VSV hyperplane

Method 2: Kernel Jittering

- nice name for (simple?) concept:
  take virtual example with smallest distance
- linear factor in run-time (vs. quadratic for VSV)

- triangular inequality?
- efficiency:
  cache reuse, SMO algorithm (sequential minimal optimization)

Results (USPS)

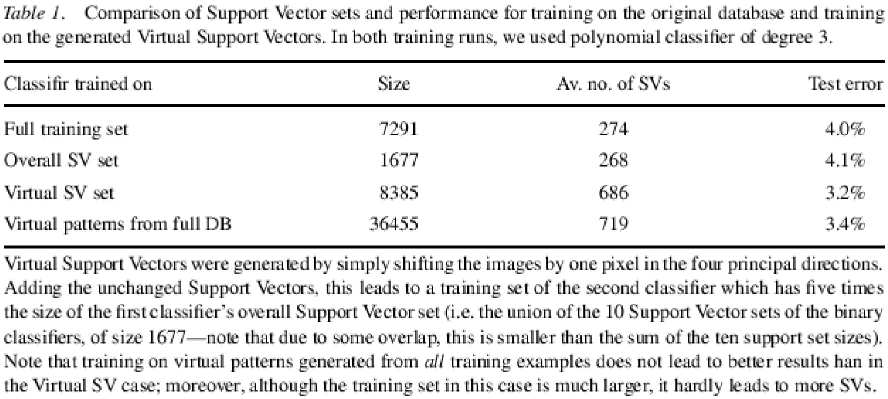


Table 1. Comparison of Support Vector sets and performance for training on the original database and training on the generated Virtual Support Vectors. In both training runs, we used polynomial classifier of degree 3.

Classifir trained on	Size	Av. no. of SVs	Test error
Full training set	7291	274	4.0%
Overall SV set	1677	268	4.1%
Virtual SV set	8385	686	3.2%
Virtual patterns from full DB	36455	719	3.4%

Virtual Support Vectors were generated by simply shifting the images by one pixel in the four principal directions. Adding the unchanged Support Vectors, this leads to a training set of the second classifier which has five times the size of the first classifier's overall Support Vector set (i.e. the union of the 10 Support Vector sets of the binary classifiers, of size 1677—note that due to some overlap, this is smaller than the sum of the ten support set sizes). Note that training on virtual patterns generated from *all* training examples does not lead to better results han in the Virtual SV case; moreover, although the training set in this case is much larger, it hardly leads to more SVs.

Results (USPS)

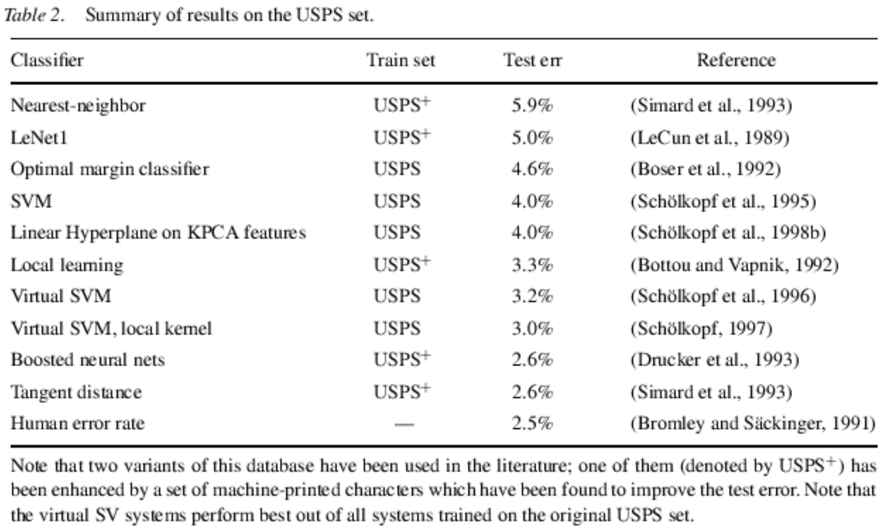


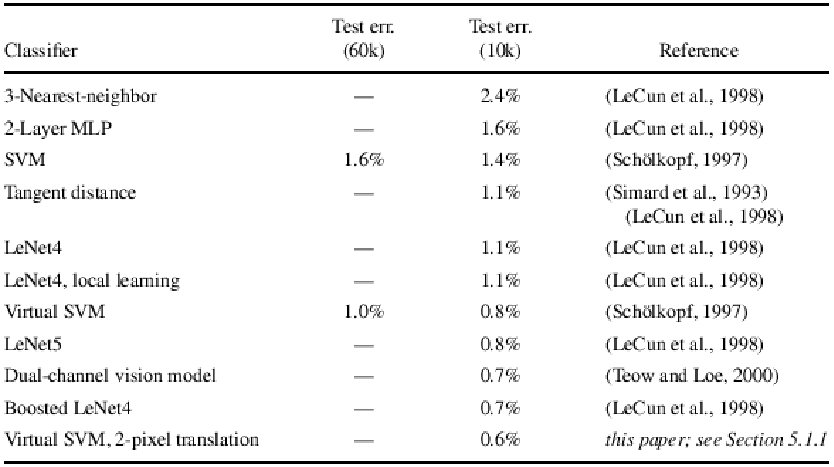
Table 2. Summary of results on the USPS set.

Classifier	Train set	Test err	Reference
Nearest-neighbor	USPS$^+$	5.9%	(Simard et al., 1993)
LeNet1	USPS$^+$	5.0%	(LeCun et al., 1989)
Optimal margin classifier	USPS	4.6%	(Boser et al., 1992)
SVM	USPS	4.0%	(Schölkopf et al., 1995)
Linear Hyperplane on KPCA features	USPS	4.0%	(Schölkopf et al., 1998b)
Local learning	USPS$^+$	3.3%	(Bottou and Vapnik, 1992)
Virtual SVM	USPS	3.2%	(Schölkopf et al., 1996)
Virtual SVM, local kernel	USPS	3.0%	(Schölkopf, 1997)
Boosted neural nets	USPS$^+$	2.6%	(Drucker et al., 1993)
Tangent distance	USPS$^+$	2.6%	(Simard et al., 1993)
Human error rate	—	2.5%	(Bromley and Säckinger, 1991)

Note that two variants of this database have been used in the literature; one of them (denoted by USPS$^+$) has been enhanced by a set of machine-printed characters which have been found to improve the test error. Note that the virtual SV systems perform best out of all systems trained on the original USPS set.

Results (MNIST)

Table 3. Summary of results on the MNIST set. At 0.6% (0.56% before rounding), the system described in Section 5.1.1 performs best.



Classifier	Test err. (60k)	Test err. (10k)	Reference
3-Nearest-neighbor	—	2.4%	(LeCun et al., 1998)
2-Layer MLP	—	1.6%	(LeCun et al., 1998)
SVM	1.6%	1.4%	(Schölkopf, 1997)
Tangent distance	—	1.1%	(Simard et al., 1993) (LeCun et al., 1998)
LeNet4	—	1.1%	(LeCun et al., 1998)
LeNet4, local learning	—	1.1%	(LeCun et al., 1998)
Virtual SVM	1.0%	0.8%	(Schölkopf, 1997)
LeNet5	—	0.8%	(LeCun et al., 1998)
Dual-channel vision model	—	0.7%	(Teow and Loe, 2000)
Boosted LeNet4	—	0.7%	(LeCun et al., 1998)
Virtual SVM, 2-pixel translation	—	0.6%	*this paper; see Section 5.1.1*

MNIST (deslant): 1.22% SV / 0.68% VSV / 0.56% VSV2

SVM vs Neural Net

“It should be noted that while it is much slower in training, the LeNet4 ensemble also has the advantage of a faster runtime speed. Especially when the number of SVs is large, SVMs tend to be slower at runtime than neural networks of comparable capacity. This is particularly so for virtual SV systems, which work by increasing the number of SV.”

Pros and Cons

pros and cons of SVMs (personal view)

pro:

- nice tools available
- state-of-the-art method
- nice theoretical basis

cons:

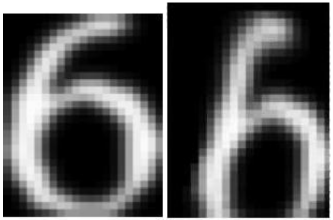
- classification $\neq$ SVM
- problems with many classes
- can be very slow

conclusions for this approach:

- incorporating prior knowledge about invariances into SVMs
- state-of-the-art performance
- there are many alternatives
- "the best neural networks [...] still appear to be much faster at test time than our best SVMs"

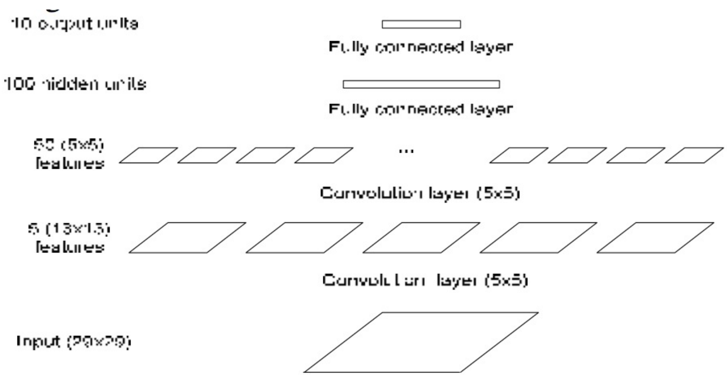
Convolutional Neural Net (Simard et al.)

Simard & Steinkraus$^+$ 2003 (MSR)
generate large amount of virtual data on the fly ($\sim$factor 1000)
during training of a well-designed neural network



excellent results (see tables later)

Convolutional Neural Network



by sharing weights, the first layers act as a trained feature extractor

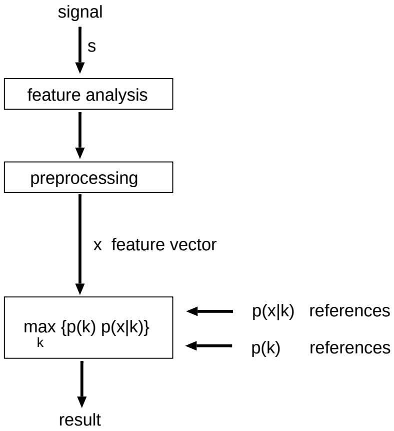
Statistical Approach

goal:
minimize the decision errors
$\rightarrow$ Bayes decision rule:

$$\arg\max_k p(k|x) = \\ = \arg\max_k \{p(k) \cdot p(x|k)\}$$

holistic image recognition:
– no segmentation
– feature vector = pixels
appearance–based approach

invariance can be tackled in
– feature analysis (invariant features)
– preprocessing (normalization)
– references (virtual data)
– $p(x|k)$ (invariant p.d.f./distance)



Decision Rule

Bayes' rule:

$$r(x) = \arg\max_k \{p(k)p(x|k)\}$$

Gaussian mixtures:

$$r(x) = \arg\max_k \left\{ p(k) \sum_i c_i \, \mathcal{N}(x|\Sigma_{ki}, \mu_{ki}) \right\}$$

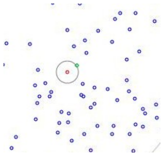
kernel densities:

$$r(x) = \arg\max_k \left\{ \sum_n \mathcal{N}(x|\Sigma_{kn}, \mu_{kn}) \right\}$$

Decision Rule

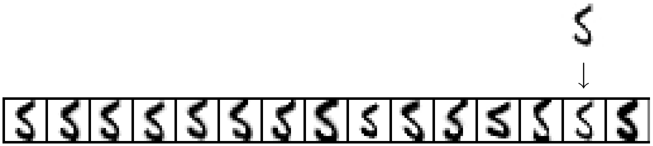
nearest neighbor decision rule:

$$r(x) = \arg\min_k \left\{ \min_n \ d(x, \mu_{kn}) \right\}$$



use invariant distance measure of the general form:

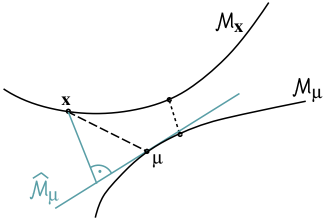
$$d(x, \mu) = \min_\alpha \left\{ d'\big(x, t(\mu, \alpha)\big) \right\}$$



Linear Matching: Tangent Distance

introduced by Simard$^+$ in 1993

transformation $t(x, \alpha) \rightarrow$ manifold
$\mathcal{M}_x = \{ t(x, \alpha) \ : \ \alpha \in R^L \} \subset R^D$



manifold distance $d(x, \mu) = \min\limits_{\alpha, \beta \in R^L} \{ ||t(x, \alpha) - t(\mu, \beta)||^2 \}$

hard optimization problem $\rightarrow$ linear approximation to transformation t:
subspace spanned by the tangent vectors

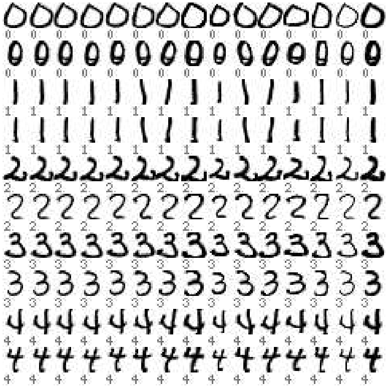
$$v_l = \frac{\partial t(\mu, \alpha)}{\partial \alpha_l}$$

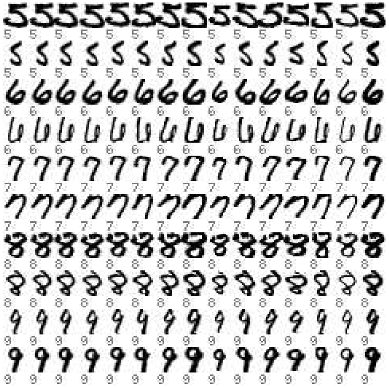
$$\widehat{\mathcal{M}}_\mu = \{ \mu + \sum_{l=1}^{L} \alpha_l v_l \ : \ \alpha \in R^L \} \subset R^D$$

one-sided tangent distance:

$$d(x, \mu) = \min_{\alpha \in R^L} \Big\{ ||x - (\mu + \sum_{l=1}^{L} \alpha_l v_l)||^2 \Big\}$$

Tangent Subspaces

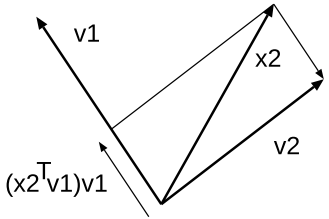




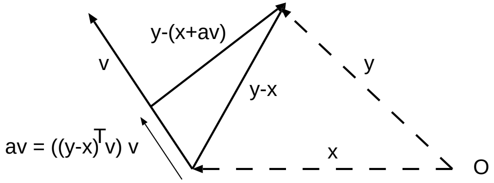
left to right:
original, 2* (vert.+ horiz.) translation, 2*rotation, 2*scale, 2*axis deformation, 2*diagonal deformation, 2*line thickness

Tangent Distance – Calculation

calculation of the distance between a point and a linear subspace
different possibilities, here: use projection into orthonormal subspace
orthonormal basis $\{v_1, \dots v_L\}$:
1) basis of subspace
2) $v_i^T v_j = \delta(i,j) = 1$ if $i = j$ and 0 otherwise
determine orthonormal basis
here (exercises): Gram-Schmidt orthogonalization and normalize
$\{x_1, \dots x_L\} \to$ orthonormal basis $\{v_1, \dots v_L\}$
1) $v_1 \leftarrow \frac{1}{||x_1||} x_1$ (one vector is always orthogonal)
2) $v_2 \leftarrow x_2 - (x_2^T v_1) v_1; \qquad v_2 \leftarrow \frac{1}{||v_2||} v_2 \qquad (a^T b = ||a||\,||b|| \cos(\gamma))$
...
$l)\ v_l \leftarrow x_l - \sum_{n=1}^{l-1} (x_l^T v_n) v_n; \qquad v_l \leftarrow \frac{1}{||v_l||} v_l$
...
linear independence assumed
what happens otherwise?



Tangent Distance – Calculation



$$\begin{aligned}
||y-(x+av)||^2 &= ||y-(x+((y-x)^Tv)v)||^2\\
&= ||(y-x)-((y-x)^Tv)v||^2\\
&= ||y-x||^2-||((y-x)^Tv)v||^2\\
&= ||y-x||^2-||(y-x)^Tv||^2\,||v||^2\\
&= ||y-x||^2-||(y-x)^Tv||^2
\end{aligned}$$

You can use any of these formulations. This needs to be extended to a subspace of higher dimension. (Easy because of orthonormal representation! Why?)
see the difference between $x^Tx = ||x||^2$ and $(x^Ty)^2 = ||x^Ty||^2$
What happens for multiple tangent vectors?

Tangent Distance – Statistical Framework

use linear subspace in statistical model

$$p(x \mid \mu, \alpha, \Sigma) = \mathcal{N}(x \mid \mu + \sum_{l=1}^{L} \alpha_l \mu_l,\ \Sigma)$$

integrate over unknown transformation parameter using

$$p(\alpha \mid \mu, \Sigma) = p(\alpha) = \mathcal{N}(\alpha \mid 0, \gamma^2 I)$$

$$\begin{aligned} p(x|\mu,\Sigma) &= \int p(x,\alpha|\mu,\Sigma)\ d\alpha \\ &= \int p(\alpha|\mu,\Sigma) \cdot p(x|\mu,\Sigma,\alpha)\ d\alpha \\ &= \int p(\alpha) \cdot p(x|\mu,\Sigma,\alpha)\ d\alpha \end{aligned}$$

Tangent Distance – Statistical Framework

result:

$$p(x|\mu,\Sigma) = \mathcal{N}(x|\mu,\Sigma') = \det(2\pi\Sigma')^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\left[(x-\mu)^T\Sigma'^{-1}(x-\mu)\right]\right)$$

$$\Sigma' = \Sigma + \gamma^2 \sum_{l=1}^{L} \mu_l \mu_l^T, \qquad \Sigma'^{-1} = \Sigma^{-1} - \frac{1}{1+\frac{1}{\gamma^2}} \Sigma^{-1} \sum_{l=1}^{L} \mu_l \mu_l^T \Sigma^{-1}$$

interpretation:

- the tangent vector approach imposes a structure on the covariance matrix
- variations along the directions of the tangent vectors are not/less important for classification

Estimation of Tangent Vectors

if the transformations are unknown
$\rightarrow$ learn the transformations from training data
$\rightarrow$ estimate the tangent vectors
log-likelihood as a function of the unknown tangent vectors $\{\mu_{kl}\}$:

$$\begin{aligned}
F(\{\mu_{kl}\}) &:= \sum_{k=1}^{K}\sum_{n=1}^{N_k} \log \mathcal{N}(x_{n,k}|\mu_k, \Sigma'_k) \\
&= \frac{1}{1+\frac{1}{\gamma^2}} \sum_{k=1}^{K}\sum_{n=1}^{N_k}\sum_{l=1}^{L} ((x_{n,k}-\mu_k)^T \Sigma^{-1}\mu_{kl})^2 + \text{const} \\
&= \frac{1}{1+\frac{1}{\gamma^2}} \sum_{k=1}^{K}\sum_{l=1}^{L} \mu_{kl}^T \Sigma^{-1} S_k \Sigma^{-1} \mu_{kl} + \text{const}
\end{aligned}$$

with $S_k = \sum_{n=1}^{N_k}(x_{n,k}-\mu_k)(x_{n,k}-\mu_k)^T$ class specific scatter matrix
result: choose $\{\mu_{kl}\}$ such that the vectors $\{\Sigma^{-1/2}\mu_{kl}\}$ are the eigenvectors with the largest corresponding eigenvalues of $\Sigma^{-1/2} S_k (\Sigma^{-1/2})^T$

Nonlinear Matching - Literature

O. Agazzi and S. Kuo. Pseudo Two-Dimensional Hidden Markov Models for Document Recognition. AT&T Technical Journal, pp. 60–72, September 1993.

S. Kuo and O. Agazzi. Keyword Spotting in Poorly Printed Documents Using Pseudo 2-D Hidden Markov Models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 16(8):842–848, August 1994.

E. Levin and R. Pieraccini. Dynamic Planar Warping for Optical Character Recognition. In ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech and Signal Processing, Vol. III, pp. 149–152, March 1992.

S. Uchida and H. Sakoe. Piecewise Linear Two-Dimensional Warping. In 15th International Conf. on Pattern Recognition, Barcelona, Spain, Vol. 3, pp. 538–541, September 2000.

Nonlinear Matching

comparing two images with flexible image planes
$\rightarrow$ allow deformation
two position dependent signals = images:

- reference image: $\mu_{xy} \in R,\ x = 1, ..., X,\ y = 1, ..., Y$
- observed image: $a_{ij} \in R,\ i = 1, ..., I,\ j = 1, ..., J$

task: find optimal image alignment

$$(i, j) \rightarrow (x, y) = (x_{ij}, y_{ij})$$

1-D signal natural to regard as
sequence $t \mapsto t + 1$ over $t = 1, ..., T - 1$
This is not the case for 2d signals.

From 1-D to 2-D

First step: consider only one axis as flexible, second axis fixed
$\rightarrow$ problem = 1-D time alignment with vector-valued signals:

$$(i,j) \rightarrow (x,y) = (x_i, j)$$

Quantitative criterion:

$$\min_{x_1^I} \left\{ \sum_{i=1}^{I} \left[\mathcal{T}(x_i - x_{i-1}) + \sum_{j=1}^{J} (\mu_{x_i j} - a_{ij})^2 \right] \right\}$$

(assumption here: $Y = J$, i.e. images of same height)

$\rightarrow$ HMM

Pseudo-2-D HMM

Now introduce flexibility in second axis:
Consider each column vector of the image as 1-D signal and use best alignment.

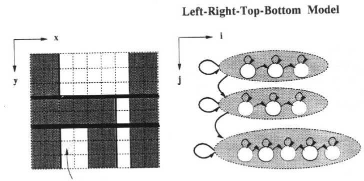
$$(i, j) \rightarrow (x, y) = (x_i, y_{ij})$$

Quantitative criterion:

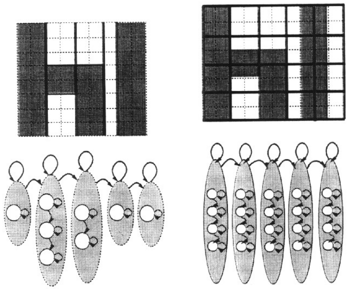
$$\min_{x_1^I} \left\{ \sum_{i=1}^{I} \left[\mathcal{T}(x_i - x_{i-1}) + \min_{y_{i1}^J} \left\{ \sum_{j=1}^{J} [\mathcal{T}(y_{ij} - y_{i,j-1}) + (\mu_{x_i y_{ij}} - a_{ij})^2] \right\} \right] \right\}$$

No interdependence between HMMs for columns assumed,
each image column is considered independently.
$\rightarrow$ called pseudo-2-D HMM

Pseudo-2-D HMM



Pseudo-2-D HMM for the word "hl"



"rotated" structures

→ independent DP for columns and rows
computationally equivalent to a 1D HMM

Pseudo-2-D HMM

application:
keyword spotting

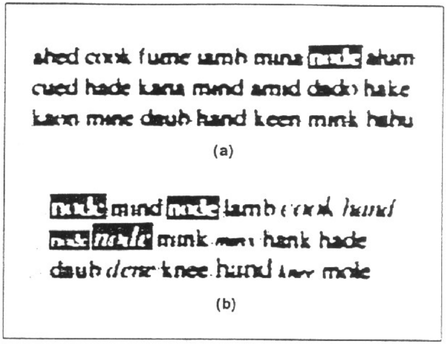

ahed cook fume lamb mina node ahum
cued hade kana mend amid dado hake
kaon mine daub hand keen mink hahu
(a)
node mind node lamb cook hand
node node mink mon hank hade
daub dcme knee hand kan mode
(b)

Figure 7. In (a), the keyword "node" is correctly spotted on this document, which contained, in one test, 2,650 key words and 16,000 extraneous words. In (b), the keyword "node" also was successfully spotted in a document that included both size and slant transformations of the word.

General Approach

test image $A = \{a_{ij}\}$ reference image $B = \{b_{xy}\}$
$a_{ij}, b_{xy} \in R^U$
image deformation mapping $(x_{11}^{IJ}, y_{11}^{IJ}) : (i,j) \mapsto (x_{ij}, y_{ij})$
mappings must fulfill constraints: $(x_{11}^{IJ}, y_{11}^{IJ}) \in \mathcal{M}$
decision rule:

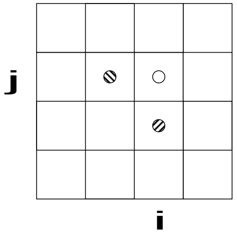
$$r(A) = \arg\min_k \Big\{ \min_{n=1,\ldots,N_k} d(A, B_{nk}) \Big\}$$

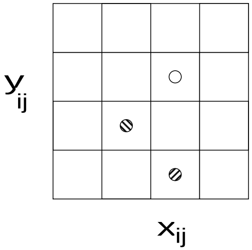
$$d(A,B) = \min_{(x_{11}^{IJ}, y_{11}^{IJ}) \in \mathcal{M}} \left\{ d'\left(A, B_{(x_{11}^{IJ}, y_{11}^{IJ})}\right) \right\}$$

$$d'\left(A, B_{(x_{11}^{IJ}, y_{11}^{IJ})}\right) = \sum_{i,j} \sum_{u} ||a_{ij}^u - b_{x_{ij}y_{ij}}^u||^2$$

2D Dependencies

image deformation mapping $(x_{11}^{IJ}, y_{11}^{IJ}) \in \mathcal{M} : (i, j) \mapsto (x_{ij}, y_{ij})$





Models for Nonlinear Matching

informal descriptions of the used models:

2DW	2-Dimensional Warping (order 2) complete 2D constraints, minimization NP-complete
P2DHMM	Pseudo 2-Dimensional Hidden Markov Model (order 1) match columns on columns, regard columns as independent
P2DHMDM	Pseudo 2-Dimensional Hidden Markov Distortion Model allow additional horizontal displacements in P2DHMM
IDM	Image Distortion Model (order 0) disregard relative displacements of neighboring pixels restrict absolute displacement

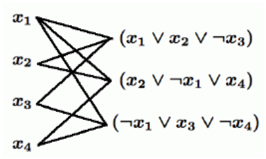
2DW Matching is NP-complete

proof idea:
Given any 3-SAT formula Φ, construct in polynomial time
an equivalent image matching problem $M(\Phi) = (A(\Phi), B(\Phi))$.
(Construct and draw dependency graph, then refine it.)

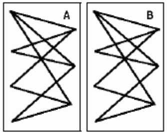
The two images of $M(\Phi)$ can be matched at cost 0 if and only if
the formula Φ is satisfiable.

$$(x_1 \vee x_2 \vee \neg x_3) \;\wedge$$
$$(x_2 \vee \neg x_1 \vee x_4) \;\wedge$$
$$(\neg x_1 \vee x_3 \vee \neg x_4)$$

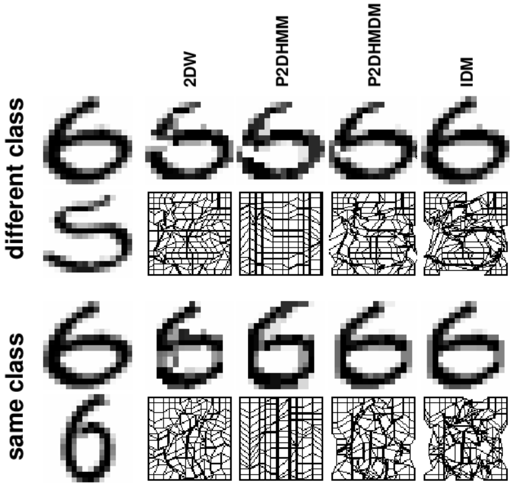
$\longrightarrow$


x_1
x_2
x_3
x_4
$(x_1 \vee x_2 \vee \neg x_3)$
$(x_2 \vee \neg x_1 \vee x_4)$
$(\neg x_1 \vee x_3 \vee \neg x_4)$

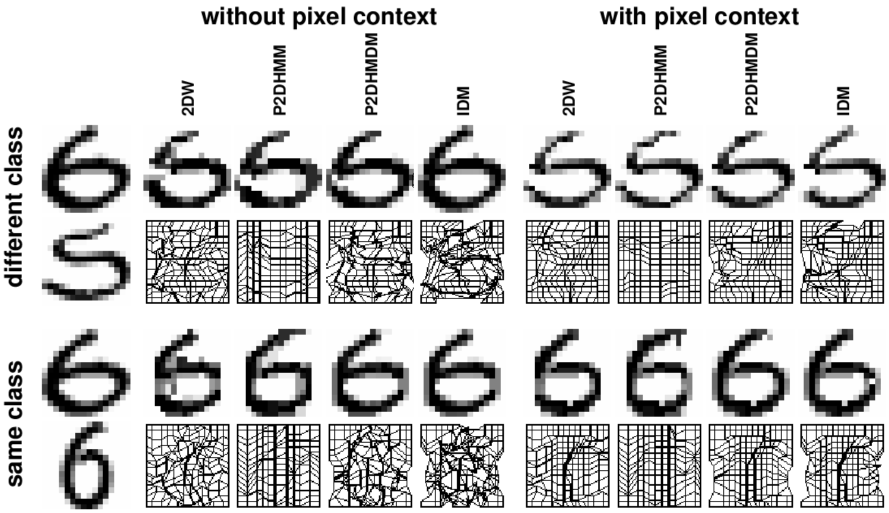
$\longrightarrow$



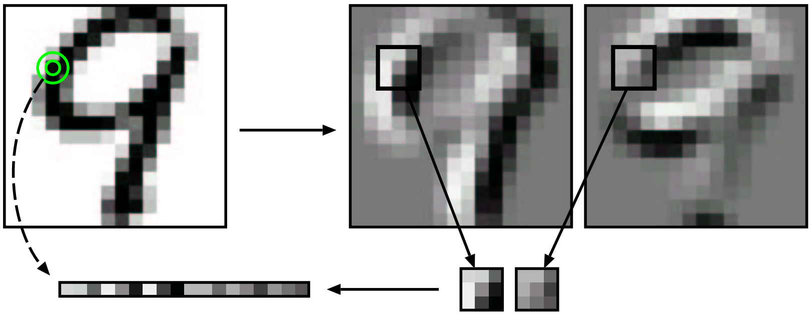
Nonlinear Matching Examples



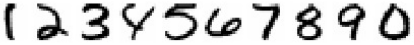
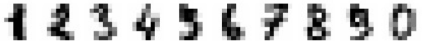
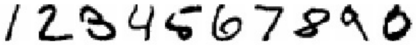
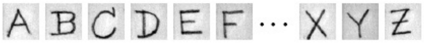
Nonlinear Matching Examples



Local Context



Data Sets

name	example images	size	# train	# test
USPS		16×16	7 291	2 007
UCI		8×8	3 823	1 797
MCEDAR		8×8	11 000	2 711
MNIST		28×28	60 000	10 000
ETL6A		64×63	15 600	13 000

all matching experiments use 3×3 context of gradients in 3-NN

USPS

name	example images	size	# train	# test
USPS		16×16	7 291	2 007

method		ER[%]
no matching, 1-NN		5.6
SVM + invariant features	ext.	3.5
invariant SVM	ext.	3.0
tangent distance	i6	3.0
2DW	i6	2.7
extended SVM training	ext.	2.5
P2DHMM	i6	2.5
tangent distance, KD, virtual data	i6	2.4
IDM	i6	2.4
extended SVM training	ext.	2.2
Hungarian matching	i6	2.2
local patches + tangent distance	i6	2.0
P2DHMDM	i6	1.9

MCEDAR

name	example images	size	# train	# test
MCEDAR		8×8	11 000	2 711

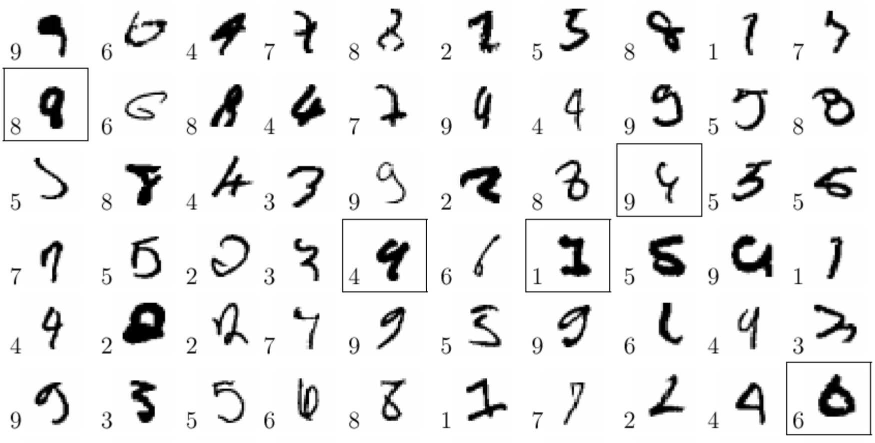
method		ER[%]
no matching, 1-NN		5.7
PCA	ext.	4.9
Bayesian PCA	ext.	4.8
factor analysis	ext.	4.7
probabilistic PCA	ext.	4.6
local patches	i6	4.3
IDM	i6	3.5
P2DHMDM	i6	3.3

MNIST

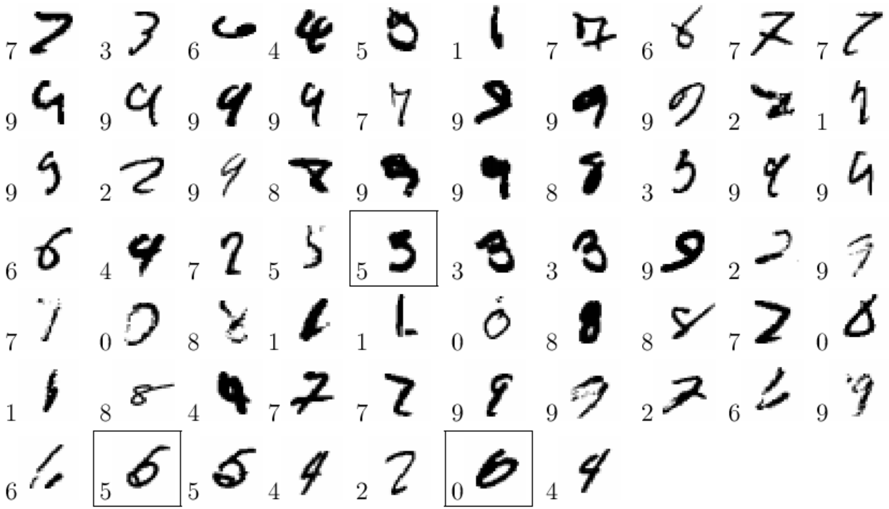
name	example images	size	# train	# test
MNIST		28×28	60 000	10 000

method		ER[%]
no matching, 1-NN		3.1
tangent distance, KD, virtual data	i6	1.0
biol. inspired features, SVM	ext.	0.72
invariant SVM	ext.	0.68
shape context matching, 3-NN (∗)	ext.	0.63
extended SVM training	ext.	0.60
biol. inspired features	ext.	0.59
invariant SVM (∗)	ext.	0.56
IDM (∗)	i6	0.54
P2DHMDM	i6	0.52
preprocessing, SVM	ext.	0.42
distortions+, neural net (∗)	ext.	0.42
combination of (∗)	i6	0.35

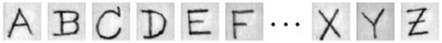
MNIST errors



MNIST errors



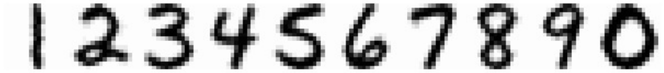
ETL6-A

name	example images	size	# train	# test
ETL6A		64×63	15 600	13 000

method		ER[%]
no matching, 1-NN		4.5
preprocessing, 1-NN	ext.	1.9
Eigen-deformations	ext.	1.1
piece-wise linear 2D-HMM	ext.	0.9
Eigen-deformations	ext.	0.8
Eigen-deformations	ext.	0.6
Eigen-deformations	ext.	0.5
IDM	i6	0.5

Learning Prototypes

means without deformation



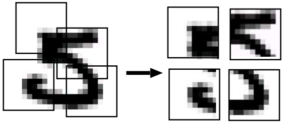
means with deformation

USPS error rates [%]

distance with deformation	deformation in mean calculation	
	no	yes
no	18.6	26.1
yes	25.3	4.9

compare best results without matching for Gaussian single densities:
MMI full Σ: 5.7%   14-D est. TD subspace: 5.0%

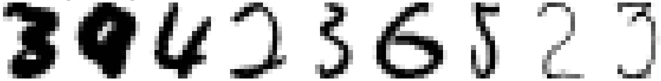
Combination of Recognition Methods

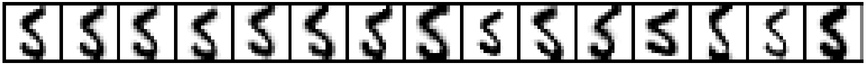


correct using local patches:

---

correct using tangents:





use classifier combination → 2.0% error on USPS