A Support Vector Method for Multivariate Performance Measures

Thorsten Joachims

Cornell University

Department of Computer Science

Thanks to

Rich Caruana, Alexandru Niculescu-Mizil, Pierre Dupont, Jérôme Callut

Supervised Learning

- **Find function from input space *X* to output space *Y***

$$h : X \longrightarrow \{+1, -1\}$$

such that the ~~prediction error~~ is low.

Text Classification:
- F_1-Score
- Precision/Recall Break-Even (PRBEP)

Medical Diagnosis:
- ROC Area

Information Retrieval:
- Precision at 10

Related Work

- **Approach “Estimate Probabilities”**
  - E.g. [Platt, 2000] [Langford & Zadrozny, 2005] [Niculescu-Mizil & Caruana, 2005]
  - Potentially solve harder problem than required
- **Approach “Optimize Substitute Loss, then Post-Process”**
  - E.g. [Lewis, 2001] [Yang, 2001] [Abe et al. 2004] [Caruana & Niculescu-Mizil, 2004]
  - Typically multi-step approach, cross-validation
- **Approach “Directly Optimize Desired Loss”**
  - Linear cost models: e.g. [Morik et al., 1999] [Lin et al., 2002]
  - ROC-Area: e.g. [Herbrich et al. 2000] [Rakotomamonjy, 2004] [Cortes & Mohri, 2003] [Freund et al., 1998] [Yan et al., 2003] [Ferri et al., 2002]
  - F_1-Score: difficult [Musicant et al. 2003]

Overview

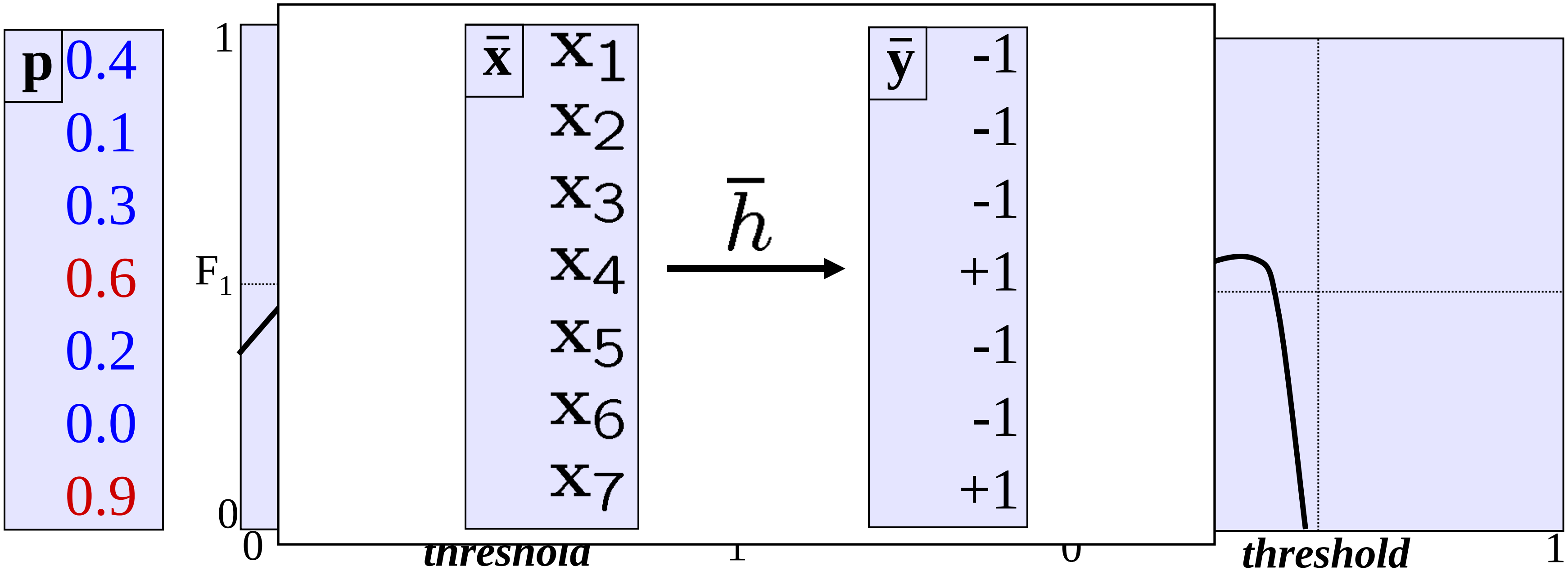
- **Formulation of Support Vector Machine for**
  - any loss function that can be computed from the contingency table.
    - F1-score, Error Rate, Linear Cost Models, etc.
  - any loss function that can be computed from contingency tables with cardinality constraints.
    - PRBEP, Prec@k, Rec@k, etc.
  - ROC-Area
- **Polynomial Time Algorithm**
- **Conventional classification SVM is special case**
  - New optimization problem
  - New representation and (extremely sparse) support vectors

Optimizing F_1 -Score

- **F_1 -score is non-linear function of example set**
  - F_1-score: harmonic average of precision and recall

$$F_1 = \frac{2\, Prec\, Rec}{Prec + Rec}$$

  - For example vector $\mathbf{x_1}$. Predict $y_1=1$, if $P(y_1=1|\mathbf{x_1})=0.4$?
    ➔ Depends on other examples!



Approach: Multivariate Prediction

- **Training Data:** $S = ((\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)) \sim_{i.i.d} \mathsf{Pr}(\mathbf{x}, y)$
- **Conventional Setting:** learn $h : X \longrightarrow \{-1, +1\}$

$$R^{\delta}(h) = \int \delta\left(h(\mathbf{x}'), y'\right) d\,\mathsf{Pr}(\mathbf{x}', y')$$

$$\hat{R}^{\delta}_{S}(h) = \frac{1}{n} \sum_{i=1}^{n} \delta\left(h(\mathbf{x}_i), y_i\right)$$

- **Multivariate Setting:** learn $\bar{h} : X^n \longrightarrow \{-1, +1\}^n$

$$R^{\Delta}(\bar{h}) = \int \Delta\left(\bar{h}(\mathbf{x}'_1, \ldots, \mathbf{x}'_{n'}), (y'_1, \ldots, y'_{n'})\right) d\,\mathsf{Pr}(S')$$

$$\hat{R}^{\Delta}_{S}(\bar{h}) = \Delta\left(\bar{h}(\mathbf{x}_1, ..., \mathbf{x}_n), (y_1, ..., y_n)\right)$$

Note:
If $\Delta\left(\bar{h}(\mathbf{x}_1, \ldots, \mathbf{x}_n), (y_1, \ldots, y_n)\right) = \frac{1}{n} \sum_{i=1}^{n} \delta\left(h(\mathbf{x}_i), y_i\right)$
then both settings are equivalent.

Support Vector Machine [Vapnik et al.]

- **Training Examples:** $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n) \quad \mathbf{x} \in \Re^N \quad y \in \{+1, -1\}$
- **Hypothesis Space:** $h(\mathbf{x}) = sgn\left[\mathbf{w}^T\mathbf{x} + b\right]$ with $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$
- **Training:** Find hyperplane $<\mathbf{w}, b>$ with minimal $\frac{1}{\delta^2} + C \sum_{i=1}^{n} \xi_i$

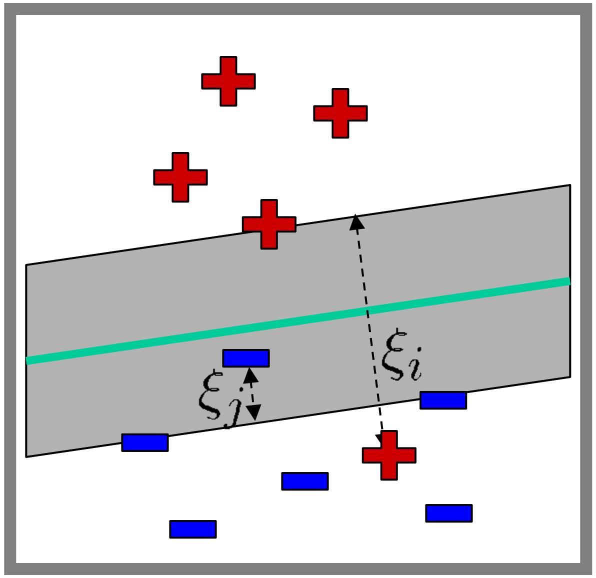
Optimization Problem:

$$\min_{\mathbf{w},\xi,b} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C \sum_{i=1}^{n} \xi_i$$

$$s.t. \quad y_1(\mathbf{w}^T\mathbf{x}_1 + b) \geq 1 - \xi_1$$

$$\ldots$$

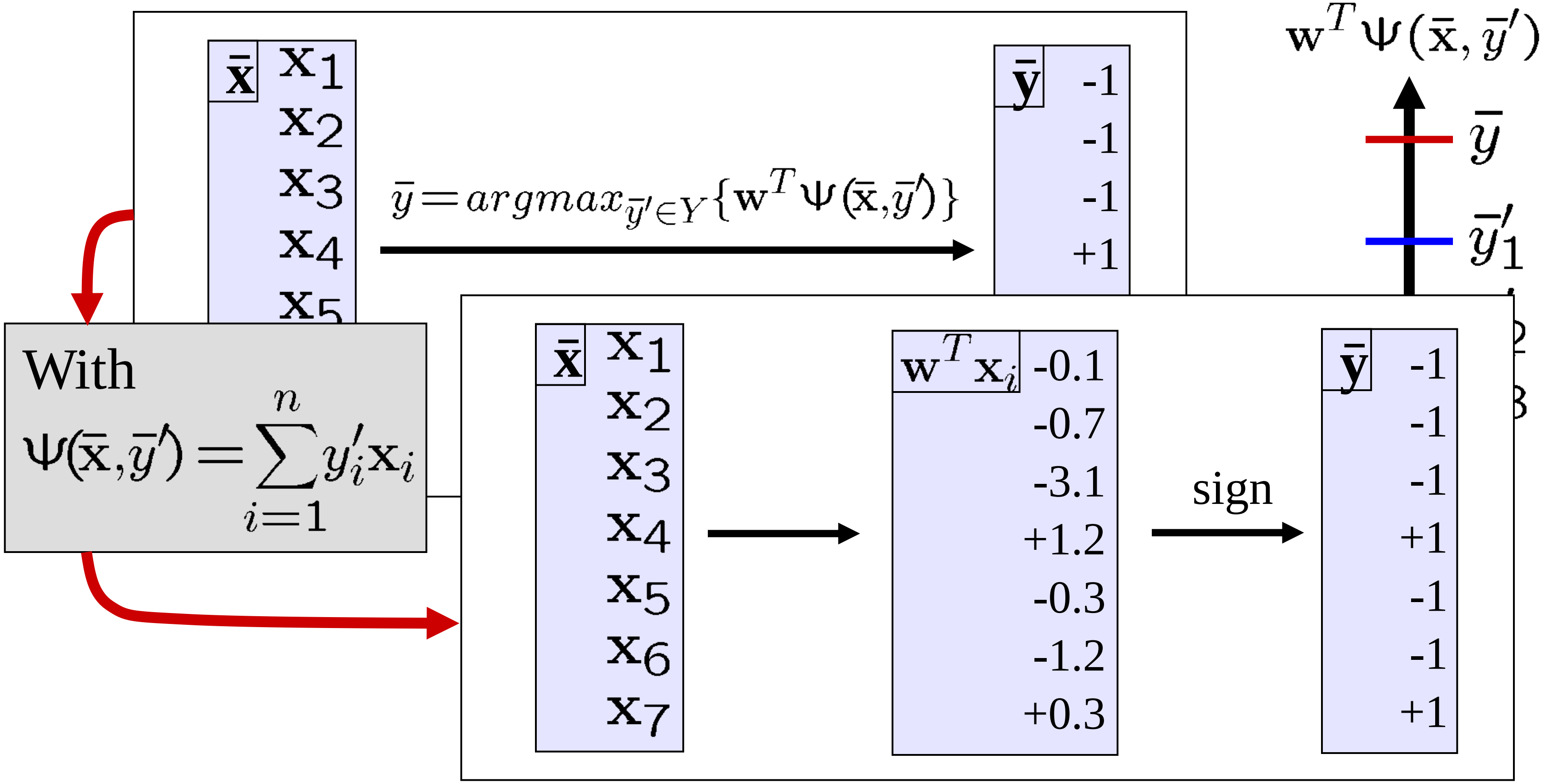
$$y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1 - \xi_n$$



Multivariate Support Vector Machine

Approach: Linear Discriminant [Collins 2002] [Lafferty et al. 2002] [Taskar et al. 2004] [Tsochantaridis et al. 2004] etc.

– "Learn weights $\mathbf{w}$ so that $\mathbf{w}^T\Psi(\bar{\mathbf{x}},\bar{y})$ is max for correct $\bar{y}$."



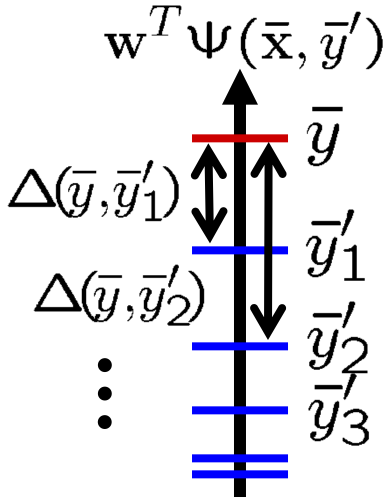
Multivariate SVM Optimization Problem

Approach: Structural SVM [Taskar et al. 04] [Tsochantaridis et al. 04]

Hard-margin optimization problem:

$$\min_{\mathbf{w}} \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

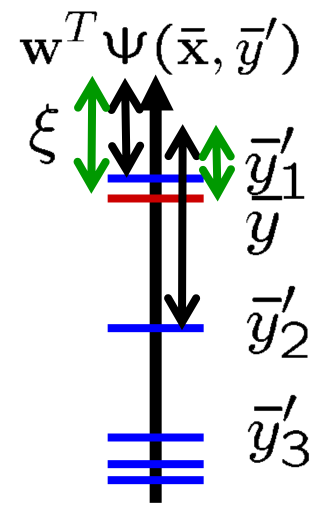
$$s.t.\ \forall \bar{y}' \in \bar{\mathcal{Y}} \setminus \{\bar{y}\} : \mathbf{w}^T\Psi(\bar{\mathbf{x}}, \bar{y}) \geq \mathbf{w}^T\Psi(\bar{\mathbf{x}}, \bar{y}') + 1$$



Soft-margin optimization problem:

$$\min_{\mathbf{w},\xi\geq 0} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\xi$$

$$s.t.\ \forall \bar{y}' \in \bar{\mathcal{Y}} : \mathbf{w}^T\Psi(\bar{\mathbf{x}}, \bar{y}) \geq \mathbf{w}^T\Psi(\bar{\mathbf{x}}, \bar{y}') + \Delta(\bar{y}, \bar{y}') - \xi$$



Theorem: At the solution, the training loss is upper bounded by $\Delta(\bar{y}, \bar{h}(\bar{\mathbf{x}})) \leq \xi$.

Multivariate SVM Generalizes Classification SVM

Theorem: The solutions of the multivariate SVM with number of errors as the loss function and an (unbiased) classification SVM are equal.

Multivariate SVM optimizing Error Rate:

$$\min_{\mathbf{w},\xi\geq 0} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\xi$$

$$s.t.\ \forall \bar{y}' \in \bar{\mathcal{Y}} : \mathbf{w}^T\Psi(\bar{\mathbf{x}},\bar{y}) \geq \mathbf{w}^T\Psi(\bar{\mathbf{x}},\bar{y}') + 2Err(\bar{y},\bar{y}') - \xi$$

$$\Leftrightarrow$$

Classification SVM (unbiased):

$$\min_{\mathbf{w},\xi\geq 0} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + 2C\sum_{i=1}^{n}\xi_i$$

$$s.t. \quad y_1(\mathbf{w}^T\mathbf{x}_1) \geq 1-\xi_1, \quad \ldots \quad , \quad y_n(\mathbf{w}^T\mathbf{x}_n) \geq 1-\xi_n$$

Cutting Plane Algorithm for Multivariate SVM

Approach: Sparse Approx. Structural SVM [Tsochantaridis et al. 04]

- **Input:** $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_n), \bar{y} = (y_1, \ldots, y_n), C, \epsilon$
- $S \leftarrow \emptyset, \mathbf{w} \leftarrow 0, \xi \leftarrow 0$
- **REPEAT**
  - compute $\bar{y}' = argmax_{\bar{y}' \in Y}\{\Delta(\bar{y},\bar{y}') + \mathbf{w}^T\Psi(\bar{\mathbf{x}},\bar{y}')\}$
  - IF $(\Delta(\bar{y}_i, \bar{y}') - \mathbf{w}^T[\Psi(\bar{\mathbf{x}},\bar{y}) - \Psi(\bar{\mathbf{x}},\bar{y}')]) > \xi + \epsilon$
    - $S \leftarrow S \cup \{\mathbf{w}^T[\Psi(\bar{\mathbf{x}},\bar{y}) - \Psi(\bar{\mathbf{x}},\bar{y}')] \geq \Delta(\bar{y}, \bar{y}') - \xi\}$
    - $[\mathbf{w}, \xi] \leftarrow$ optimize SVM objective over S
  - ENDIF
- **UNTIL S has not changed during iteration**

Find most violated constraint

Violated by more than ε ?

Add constraint to working set

Polynomial Convergence Bound

- **Theorem [Tsochantaridis et al., 2004]:** The sparse-approximation algorithm finds a solution to the soft-margin optimization problem after adding at most

$$\max\left\{\frac{2L}{\epsilon}, \frac{8Cn^2R^2L}{\epsilon^2}\right\}$$

constraints to the working set S, so that the Kuhn-Tucker conditions are fulfilled up to a precision ϵ. The loss has to be bounded $0 \leq \Delta(\bar{y}, \bar{y}') \leq L$, and $R = \max_i ||\mathbf{x}_i||$.

ARGMAX for Contingency Table

- **Problem:**
  - $\underset{\bar{y}' \in \{1,-1\}^n}{\text{argmax}} \left\{ \Delta(\bar{y}, \bar{y}') + \mathbf{w}^T \sum_{i=1}^{n} y_i' \mathbf{x}_i \right\}$
- **Key Insight:**
  - Only n^2 different contingency tables exist.
  - ARGMAX for each table easy to compute via sorting.
  - Time $O(n^2)$
- **Applies to:**
  - Errorrate, F_1, Prec@k, Rec@k, PRBEP, etc.

Input: $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, $\bar{y} = (y_1, \ldots, y_n)$, $\bar{\mathcal{Y}}$
$(i_1^p, \ldots, i_{\#pos}^p) \leftarrow$ sort$\{i : y_i = 1\}$ by $\mathbf{w}^T \mathbf{x}_i$
$(i_1^n, \ldots, i_{\#neg}^n) \leftarrow$ sort$\{i : y_i = -1\}$ by $\mathbf{w}^T \mathbf{x}_i$
for $a \in [0, \ldots, \#pos]$ **do**
    $c \leftarrow \#pos - a$
    set $y'_{i_1^p}, \ldots, y'_{i_a^p}$ to 1
    set $y'_{i_{a+1}^p}, \ldots, y'_{i_{\#pos}^p}$ to -1
    for $d \in [0, \ldots, \#neg]$ **do**
        $b \leftarrow \#neg - d$
        set $y'_{i_1^n}, \ldots, y'_{i_b^n}$ to 1
        set $y'_{i_{b+1}^n}, \ldots, y'_{i_{\#neg}^n}$ to -1
        $v \leftarrow \Delta(a, b, c, d) + \mathbf{w}^T \sum_{i=1}^n y_i' \mathbf{x}_i$
        if v is the largest so far **then**
            $\bar{y}^* \leftarrow (y_1', \ldots, y_n')$
        end if
    end for
end for
return$(\bar{y}^*)$

ARGMAX for ROC-Area

- **Problem:**
  - $\underset{\bar{y}' \in \{1,-1\}^n}{\text{argmax}} \{\Delta(\bar{y},\bar{y}') + \mathbf{w}^T \sum_{i=1}^{n} y_i' \mathbf{x}_i\}$
- **Key Insight:**
  - ROC Area is proportional to "swapped pairs"
  - Loss decomposes linearly over pairs
  - Find argmax via sort in time *O(n log n)*
  - Represent n^2 pairs as

$$\Psi(\bar{\mathbf{x}}, \bar{y}) = \sum_{i=1}^{n} c_i \mathbf{x}_i \text{ with } c_i = \begin{cases} \sum_{j=1}^{\#neg} y_{ij}, & if(y_i = 1) \\ \sum_{j=1}^{\#pos} y_{ji}, & if(y_i = -1) \end{cases}$$

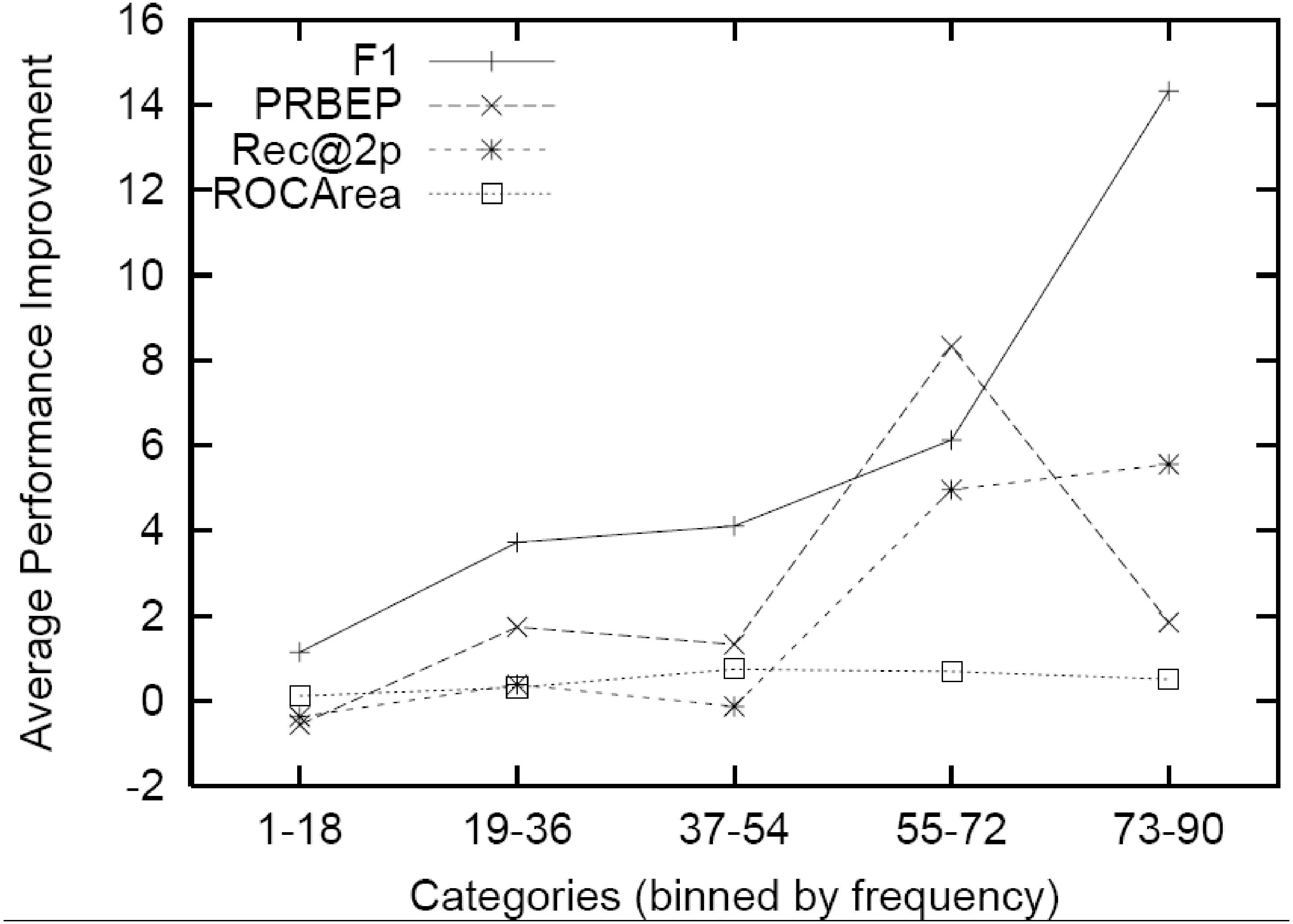
Input: $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, $\bar{y} = (y_1, \ldots, y_n)$
for $i \in \{i : y_i = 1\}$ **do** $s_i \leftarrow -0.25 + \mathbf{w}^T \mathbf{x}_i$
for $i \in \{i : y_i = -1\}$ **do** $s_i \leftarrow 0.25 + \mathbf{w}^T \mathbf{x}_i$
$(r_1, \ldots, r_n) \leftarrow$ sort $\{1, \ldots, n\}$ by s_i
$s_p = \#pos$, $s_n = 0$
for $i \in \{1, \ldots, n\}$ **do**
  if $y_{r_i} > 0$ **then**
    $c_{r_i} \leftarrow (\#neg - 2\, s_n)$
    $s_p \leftarrow s_p - 1$
  else
    $c_{r_i} \leftarrow (-\#pos + 2\, s_p)$
    $s_n \leftarrow s_n + 1$
  end if
end for
return$(c_1, \ldots, c_n)$

Experiment: Generalization Performance

- **Experiment Setup**
  - Macro-average over all classes in dataset
  - Baseline: classification SVM with linear cost model
  - Select *C* and cost ratio *j* via 2/3 – 1/3 holdout test
  - Two-tailed Wilcoxon (**=95%, *=90%)

Dataset	Method	F_1	PRBEP	$\text{Rec}_{@2p}$	ROCArea
Reuters (90 classes)	$\text{SVM}^{\Delta}_{multi}$	62.0	68.2	78.3	99.1
Examples: 9603/3299	SVM_{org}	56.1	65.7	77.2	98.6
Features: 27658	win/lose	(51/20)**	(16/8)**	(14/8)	(43/33)*
ArXiv (14 classes)	$\text{SVM}^{\Delta}_{multi}$	56.8	58.4	73.3	92.8
Examples: 1168/32487	SVM_{org}	49.6	57.9	74.4	92.7
Features: 13525	win/lose	(9/5)*	(9/4)	(1/13)**	(8/6)
Optdigits (10 classes)	$\text{SVM}^{\Delta}_{multi}$	92.5	92.7	98.4	99.4
Examples: 3823/1797	SVM_{org}	91.5	91.5	98.7	99.4
Features: 64	win/lose	(8/2)*	(5/1)*	(1/5)	(6/4)
Covertype (7 classes)	$\text{SVM}^{\Delta}_{multi}$	73.8	72.1	93.1	94.6
Examples: 1000/2000	SVM_{org}	73.9	71.0	94.7	94.1
Features: 54	win/lose	(3/4)	(5/2)	(2/5)	(4/3)

Experiment: Unbalanced Classes in Reuters



Experiment: Number of Iterations

- **Numbers averaged over**
  - all binary classification tasks within dataset
  - all parameter settings

Dataset	Method	F_1	PRBEP	$\text{Rec}_{@2p}$	ROCArea
Reuters (90 classes) Examples: 9603/3299 Features: 27658	# Iter	119.6	65.6	81.4	30.8
ArXiv (14 classes) Examples: 1168/32487 Features: 13525	# Iter	215.2	45.9	53.7	56.4
Optdigits (10 classes) Examples: 3823/1797 Features: 64	# Iter	130.5	81.5	86.8	21.6
Covertype (7 classes) Examples: 1000/2000 Features: 54	# Iter	94.3	79.8	63.7	40.2

Experiment: Number of SV

Corollary: For error rate as the loss function, the hard-margin solution after the first iteration is equal to Rocchio Algorithm.

$$\mathbf{w}_1 \sim \sum_{\{pos:y_{pos}=1\}} \mathbf{x}_{pos} - \sum_{\{neg:y_{neg}=-1\}} \mathbf{x}_{neg}$$

Dataset	Method	F_1	PRBEP	Rec$_{@2p}$	ROCArea	Err
Reuters (90 classes) Examples: 9603/3299 Features: 27658	SVM$^{\Delta}_{multi}$ SVM$_{org}$	62.0	45.0	46.3	5.1	86.3 371.4
ArXiv (14 classes) Examples: 1168/32487 Features: 13525	SVM$^{\Delta}_{multi}$ SVM$_{org}$	129.5	43.4	45.3	26.8	177.7 645.3
Optdigits (10 classes) Examples: 3823/1797 Features: 64	SVM$^{\Delta}_{multi}$ SVM$_{org}$	19.6	14.6	14.0	3.9	25.0 556.9
Covertype (7 classes) Examples: 1000/2000 Features: 54	SVM$^{\Delta}_{multi}$ SVM$_{org}$	12.5	12.0	9.4	5.0	17.1 372.8

Implementation in SVM^{struct}

- **Multivariate SVM implemented in SVM^{struct}**
  - http://svmlight.joachims.org
  - Also implementations for CFG, Sequence Alignment, OMM
- **Application specific**
  - Loss function $\Delta(\bar{y}, \bar{y}')$
  - Representation $\Psi(\bar{\mathbf{x}}, \bar{y})$
  - Algorithms to compute

$$\hat{y} = argmax_{\bar{y}' \in Y} \{ \mathbf{w}^T \Psi(\bar{\mathbf{x}}, \bar{y}') \}$$

$$\hat{y} = argmax_{\bar{y}' \in Y} \{ \Delta(\bar{y}, \bar{y}') + \mathbf{w}^T \Psi(\bar{\mathbf{x}}, \bar{y}') \}$$

$\Rightarrow$ **Generic structure that covers OMM, MPD, Finite-State Transducers, MRF, etc. (polynomial time inference)**

Conclusions

- **Generalization of SVM to multivariate loss functions**
  - Classification SVMs are special case
- **Polynomial time training algorithms for**
  - any loss function based on contingency table.
  - ROC-Area.
- **New representation of SVM optimization problem**
  - Support Vectors represent vector of classifications
  - Can be extremely sparse
- **Future work**
  - Other performance measures, other methods (e.g. boosting)
  - Faster training algorithm exploiting special structure

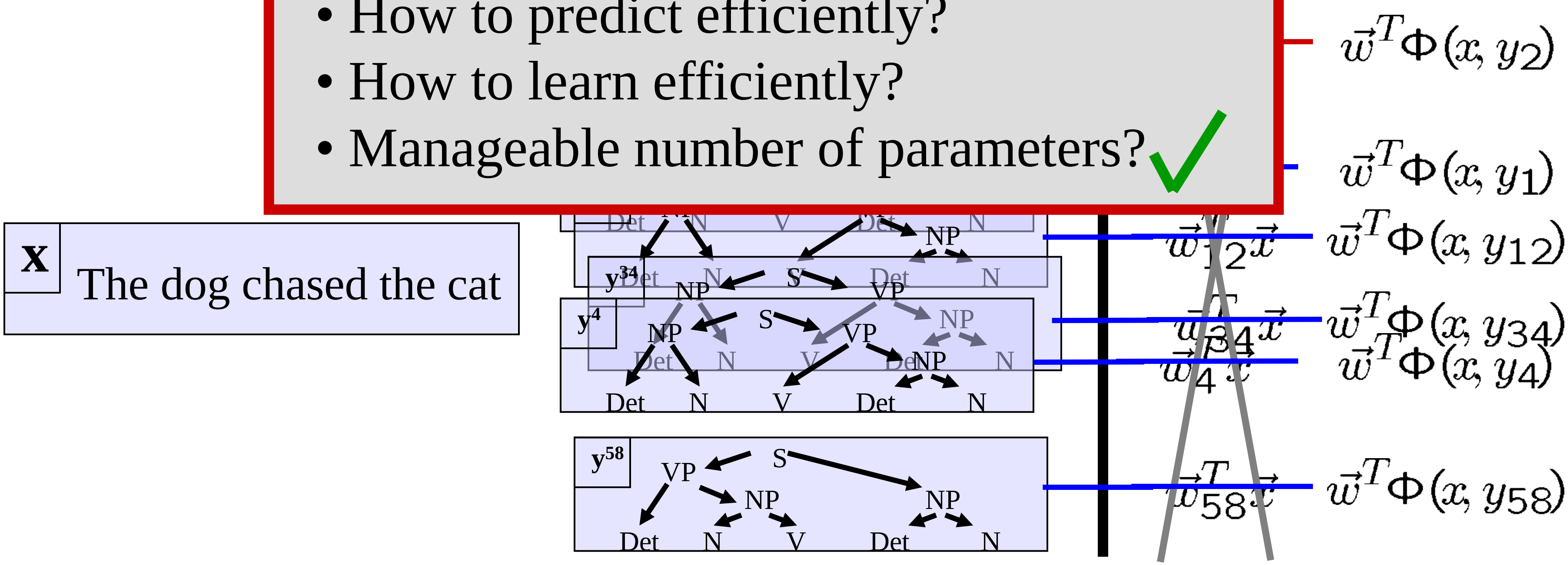
Joint Feature Map

- **Feature vector $\Phi(x, y)$ that describes match between *x* and *y***
- **Learn single weight vector and rank by $\vec{w}^T\Phi(x, y)$**

$$h(\vec{x}) = argmax_{y \in Y}\left[\vec{w}^T\Phi(x, y)\right]$$

<u>Problems</u>

- How to predict efficiently?
- How to learn efficiently?
- Manageable number of parameters? ✓



Joint Feature Map for Trees

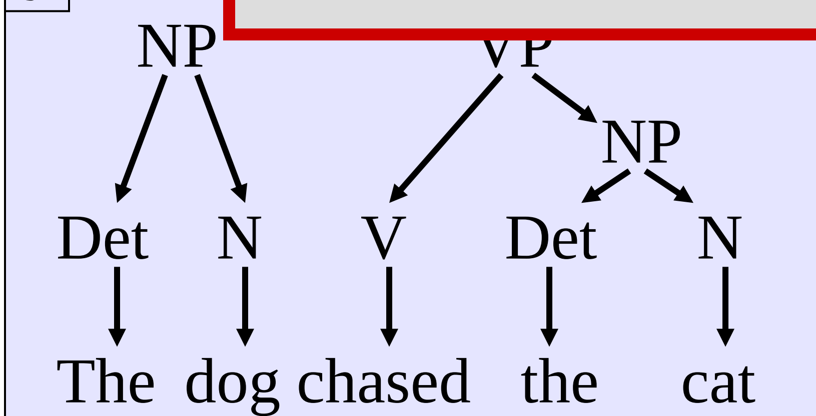
- **Weighted Context Free Grammar**
  - Each rule r_i (e.g. $S \to NP\ VP$) has a weight w_i
  - Score of a tree is the sum of its weights
  - Find highest scoring tree $h(\vec{x}) = argmax_{y \in Y} \left[\vec{w}^T \Phi(x, y) \right]$

CKY Parser

x The

$f : X$

y



Problems

- How to predict efficiently? ✓
- How to learn efficiently?
- Manageable number of parameters? ✓

$\to NP\ VP$

$\to NP$

$\to Det\ N$

$\to V\ NP$

$$\Phi(\mathbf{x}, \mathbf{y}) = \begin{pmatrix} \vdots \\ 0 \\ 2 \\ 1 \\ 1 \\ 1 \end{pmatrix} \begin{array}{l} \\ Det \to dog \\ Det \to the \\ N \to dog \\ V \to chased \\ N \to cat \end{array}$$

Experiment: Natural Language Parsing

- **Implemention**
  - Implemented Sparse-Approximation Algorithm in SVM$^{\text{light}}$
  - Incorporated modified version of Mark Johnson's CKY parser
  - Learned weighted CFG with $\epsilon = 0.01, C = 1$
- **Data**
  - Penn Treebank sentences of length at most 10 (start with POS)
  - Train on Sections 2-22: 4098 sentences
  - Test on Section 23: 163 sentences

	Test Accuracy		Training Efficiency		
Method	Acc	F_1	CPU-h	Iter	Const
PCFG with MLE	55.2	86.0	0	N/A	N/A
SVM with (1-F_1)-Loss	**58.9**	**88.5**	3.4	12	8043

More Expressive Features

- **Linear composition:** $\Phi(x, \vec{y}) = \sum_{j=1}^{k} \phi(x, y_j)$
- **So far:** $\phi(x, y_i) = \begin{pmatrix} 0 \\ \ldots \\ 0 \\ 1 \\ 0 \\ \ldots \\ 0 \end{pmatrix}$ if $y_i =$ 'S $\leftarrow$ NP VP'
- **General:** $\phi(x, y_i) = \phi_{kernel}(\phi(x, [rule, start, end]))$

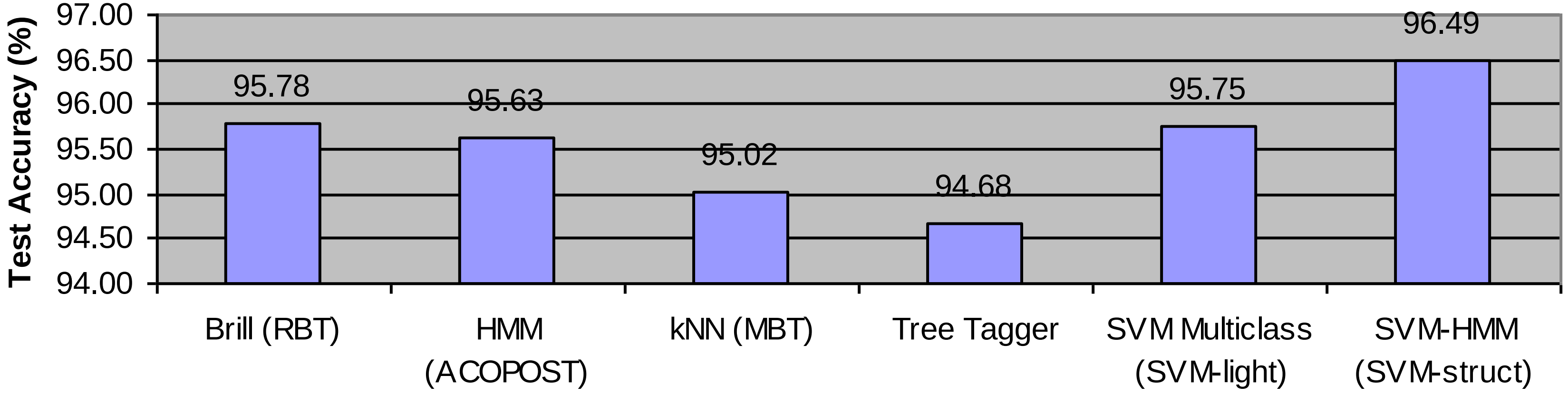
  $K(a, b) = \phi_{kernel}(a)^T \phi_{kernel}(a)$
- **Example:** $\phi(x, y_i) = \begin{pmatrix} 1 & \text{if } x_{start} = \text{'while'} \wedge x_{end} = \text{'.'} \\ (start - end)^2 & \\ 1 & \text{span contains } x_{start} = \text{'and'} \\ \ldots & \end{pmatrix}$

Experiment: Part-of-Speech Tagging

- **Task**
  - Given a sequence of words *x*, predict sequence of tags *y*.

    x The dog chased the cat → **y** Det → N → V → Det → N

  - Dependencies from tag-tag transitions in Markov model.
- **Model**
  - Markov model with one state per tag and words as emissions
  - Each word described by ~250,000 dimensional feature vector (all word suffixes/prefixes, word length, capitalization …)
- **Experiment (by Dan Fleisher)**
  - Train/test on 7966/1700 sentences from Penn Treebank



Overview

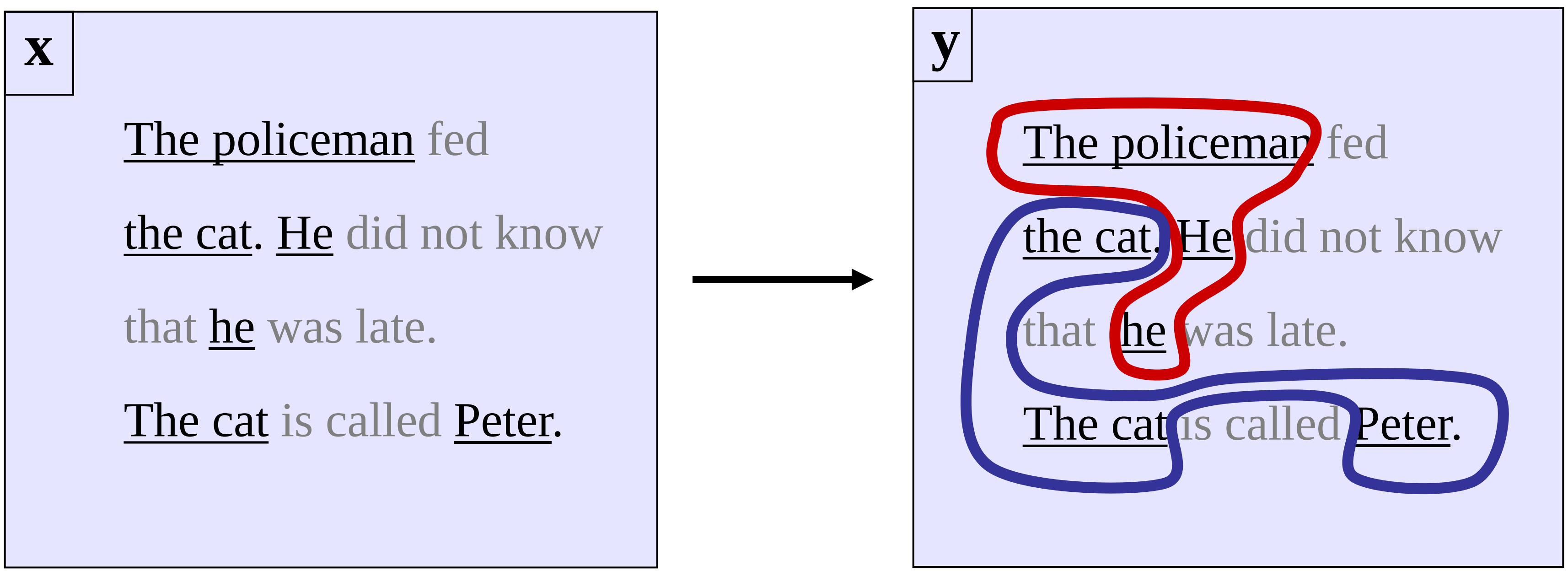
- **Task: Discriminative learning with complex outputs**
- **Related Work**
- **SVM algorithm for complex outputs**
  - Formulation as convex quadratic program
  - General algorithm
  - Sparsity bound
- **Example 1: Learning to parse natural language**
  - Learning weighted context free grammar
- **Example 2: Optimizing F_1-score in text classification**
  - Learn linear rule that directly optimizes F_1-score
- **Example 3: Learning to cluster**
  - Learning a clustering function that produces desired clusterings
- **Conclusions**

Overview

- **Task: Discriminative learning with complex outputs**
- **Related Work**
- **SVM algorithm for complex outputs**
  - Formulation as convex quadratic program
  - General algorithm
  - Sparsity bound
- **Example 1: Learning to parse natural language**
  - Learning weighted context free grammar
- **Example 2: Optimizing F_1-score in text classification**
  - Learn linear rule that directly optimizes F_1-score
- **Example 3: Learning to cluster**
  - Learning a clustering function that produces desired clusterings
- **Conclusions**

Learning to Cluster

- **Noun-Phrase Co-reference**
  - Given a set of noun phrases *x*, predict a clustering *y*.
  - Structural dependencies, since prediction has to be an equivalence relation.
  - Correlation dependencies from interactions.



Struct SVM for Supervised Clustering (by Thomas Finley)

- **Representation**
  - $\{0,1\}$
  - nd
  -
- **Los**
  - $\Delta(y, y') = ||y - y'||_1$
- **Predic**
  - $\hat{y} =$ $\sum y_{ij}(\vec{w}^T x_{ij})\}$
  - NP orlica, 2003]
- **Find n**
  - $\hat{y} =$
  - NP orlica, 2003]

$$
\begin{array}{c}
\vec{w}^T \vec{x}_i \\
\begin{array}{rrrrrrr}
1.0 & 2.5 & 0.0 & 0.0 & -2.1 & -0.1 & 0.0 \\
0.0 & 1.0 & 3.0 & 0.1 & 0.1 & 0.3 & 0.0 \\
0.0 & 0.0 & 1.0 & 0.3 & 2.2 & -0.9 & 0.0 \\
0.9 & -0.4 & -0.1 & 1.0 & -1.1 & -0.1 & -2.0 \\
0.0 & 0.0 & 0.0 & -0.3 & 1.0 & 2.3 & 0.0 \\
0.1 & 0.1 & 0.0 & -0.1 & 0.1 & 1.0 & 1.8 \\
-1.0 & -0.7 & 0.0 & -0.2 & -1.1 & 0.1 & 1.0
\end{array}
\end{array}
\longrightarrow
\begin{array}{c}
\mathbf{y} \\
\begin{array}{ccccccc}
1 & 1 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1
\end{array}
\end{array}
$$

$$
\begin{array}{c}
\mathbf{y} \\
\begin{array}{ccccccc}
1 & 1 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1
\end{array}
\end{array}
\qquad
\begin{array}{c}
\mathbf{y'} \\
\begin{array}{ccccccc}
1 & 1 & 1 & \mathbf{0} & 0 & 0 & 0 \\
1 & 1 & 1 & \mathbf{0} & 0 & 0 & 0 \\
1 & 1 & 1 & \mathbf{0} & 0 & 0 & 0 \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1
\end{array}
\end{array}
$$

Conclusions

- **Learning to predict complex output**
  - Clean and concise framework for many applications
  - Discriminative methods are feasible and offer advantages
- **An SVM method for learning with complex outputs**
- **Case Studies**
  - Learning to predict trees (natural language parsing)
  - Optimize to non-standard performance measures (imbalanced classes)
  - Learning to cluster (noun-phrase coreference resolution)
- **Software: SVM^{struct}**
  - **`http://svmlight.joachims.org/`**
- **Open questions**
  - Applications with complex outputs?
  - Is it possible to extend other algorithms to complex outputs?
  - More efficient training algorithms for special cases?

Examples of Complex Output Spaces

- **Non-Standard Performance Measures (e.g. F_1-score, Lift)**
  - F_1-score: harmonic average of precision and recall

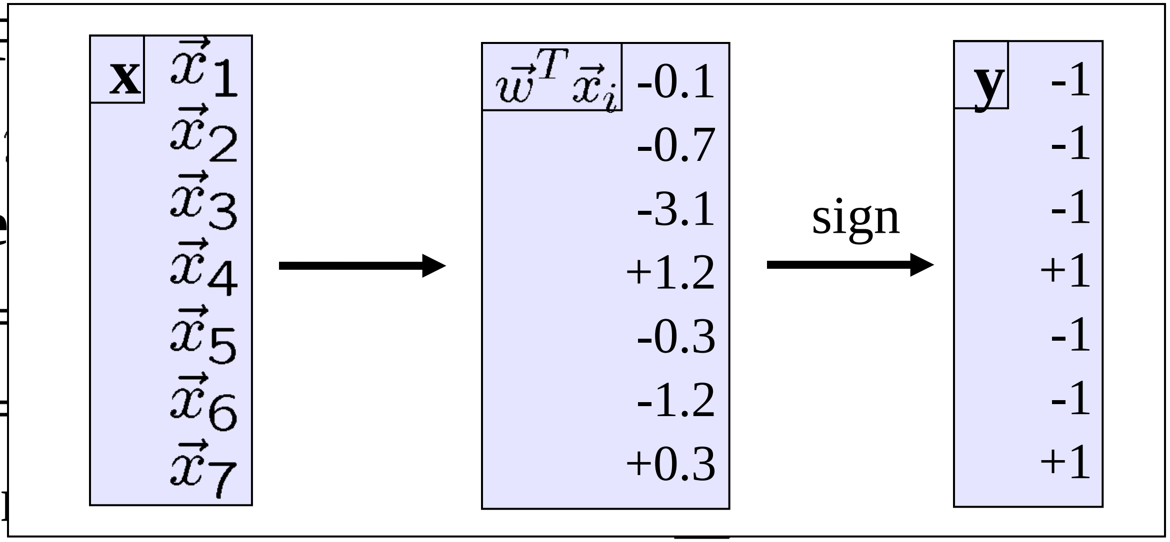
$$F_1 = \frac{2\, Prec\, Rec}{Prec + Rec}$$

  - New example vector $\vec{x}_8$. Predict $y_8=1$, if $P(y_8=1|\vec{x}_8)=0.4$?
    ➔ Depends on other examples!

x	**y**
$\vec{x}_1$	-1
$\vec{x}_2$	-1
$\vec{x}_3$	-1
$\vec{x}_4$	+1
$\vec{x}_5$	-1
$\vec{x}_6$	-1
$\vec{x}_7$	+1

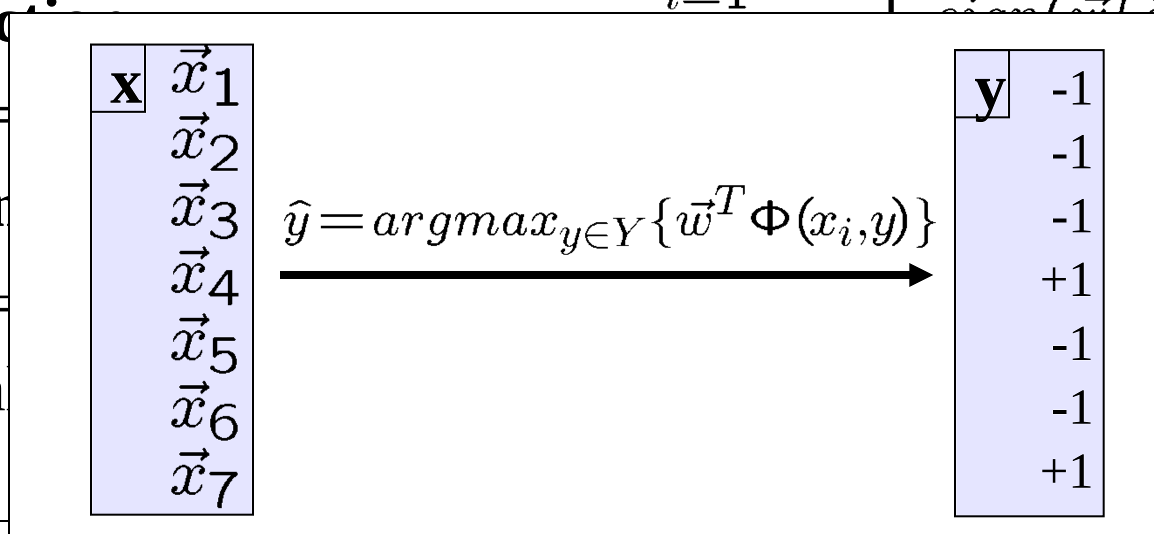
Struct SVM for Optimizing F_1-Score

- **Loss F**
  - $\Delta($
- **Repre**
  - $x =$
  - $y =$
  - Joi



$i=1$ $\left(sign(\vec{w}^T \vec{x}_1) \right)$ $\vec{x}_2)$

- **Predic**
  - $\hat{y} =$
- **Find r** $\vec{x}_n)$
  - $\hat{y} =$
  - On ... rute force



Experiment: Text Classification

- **Dataset: Reuters-21578 (ModApte)**
  - 9603 training / 3299 test examples
  - 90 categories
  - TFIDF unit vectors (no stemming, no stopword removal)
- **Experiment Setup**
  - Classification SVM with optimal C in hindsight ($C=8$)
  - F_1-loss SVM with $C=0.0625$ (via 2-fold cross-validation)
- **Results**

Method	Macroaveraged		Training Efficiency		
	Train F_1	Test F_1	CPU-min	Const	SV
Classification SVM	98.4	52.8	0.1	N/A	264
SVM with (1-F_1)-Loss	91.8	**61.8**	32.2	173	93