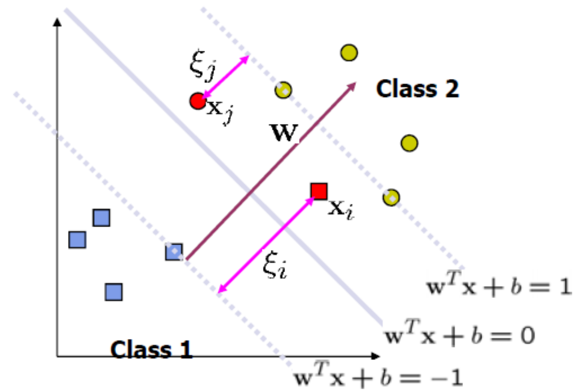
SVM: Multiclass and Structured Prediction

Bin Zhao

Part I: Multi-Class SVM

2-Class SVM



- Primal form

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

$$\text{s.t} \quad \begin{aligned} & y_i (w^T x_i + b) \geq 1 - \xi_i, \quad \forall i \\ & \xi_i \geq 0, \quad \forall i \end{aligned}$$

- Dual form

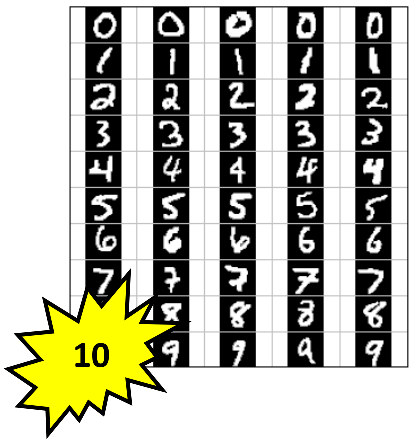
$$\max_{\alpha} \quad \mathcal{J}(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j)$$

$$\text{s.t.} \quad 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, m$$

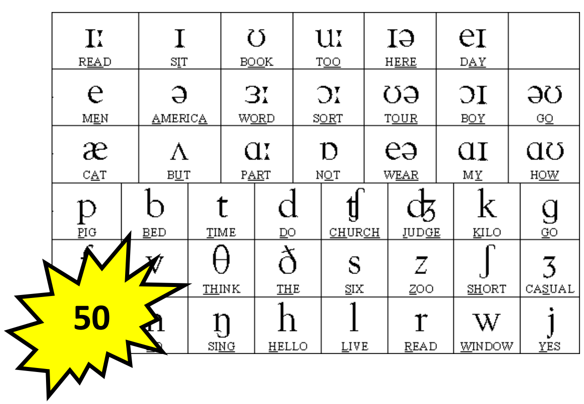
$$\sum_{i=1}^{m} \alpha_i y_i = 0.$$

Real world classification problems

Digit recognition

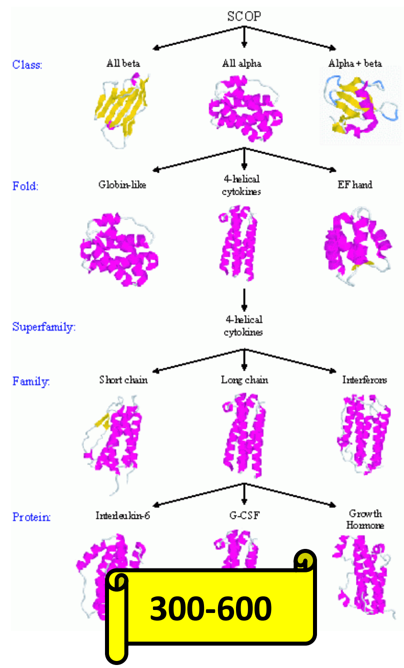


Phoneme recognition

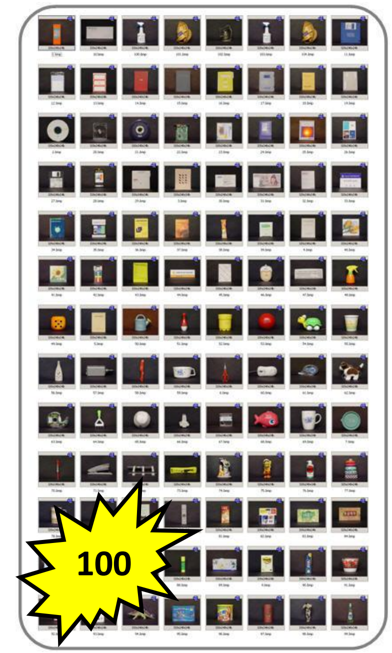


[Waibel, Hanzawa, Hinton, Shikano, Lang 1989]

Automated protein classification



Object recognition

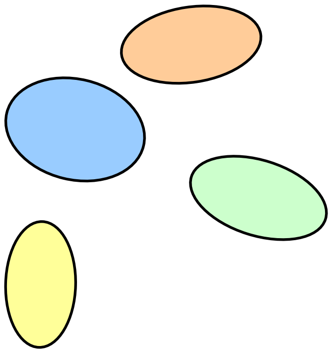


http://www.glue.umd.edu/~zhelin/recog.html

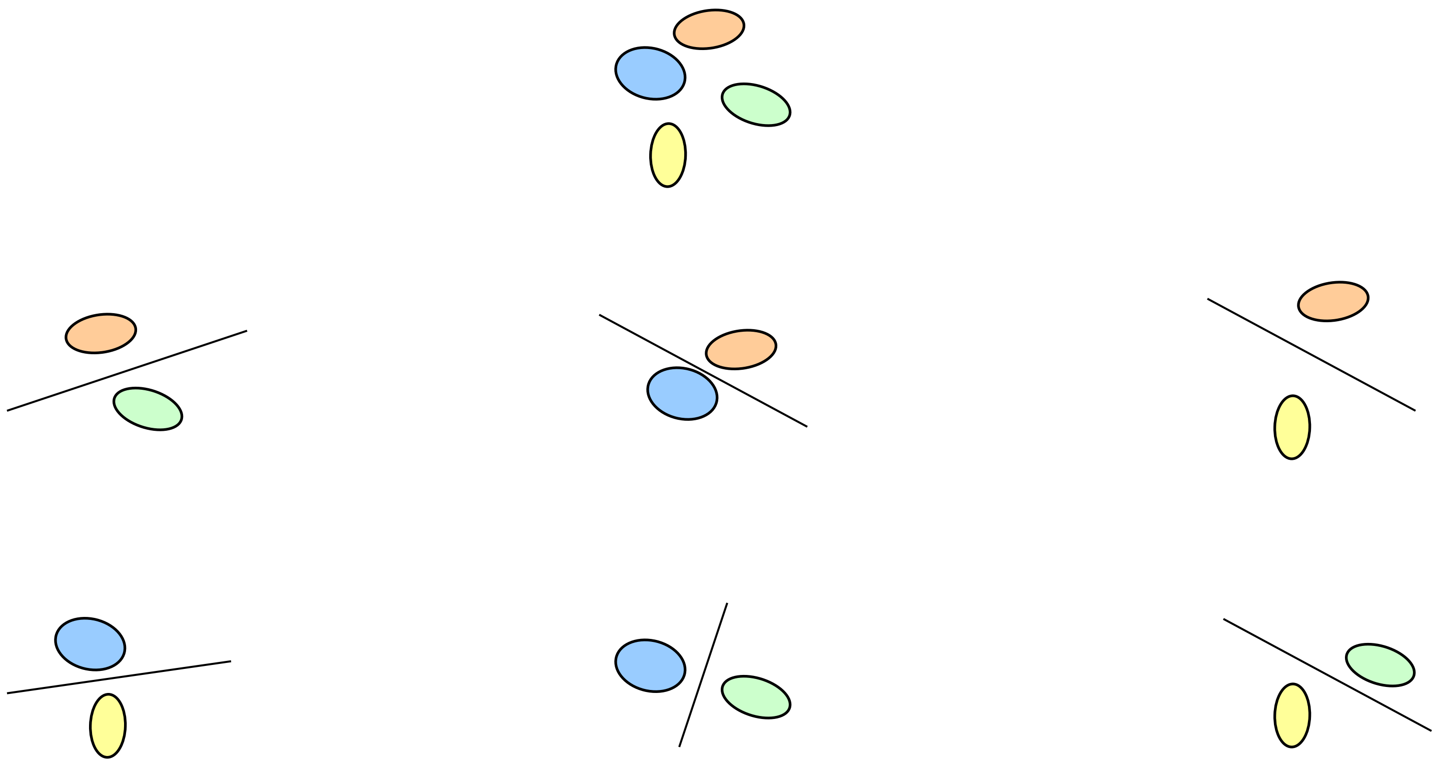
- The number of classes is sometimes big
- The multi-class algorithm can be heavy

How can we solve multi-class problem?

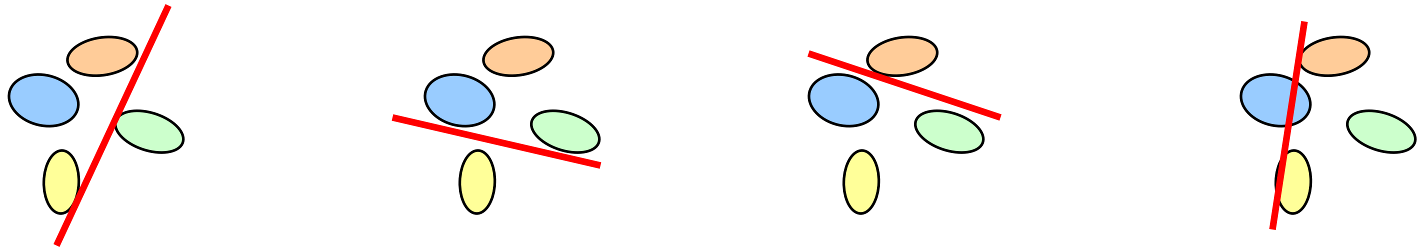
- One-against-one
- One-against-rest
- Crammer & Singer's formulation
- Error-correcting output coding
- Empirical comparisons



One-against-one

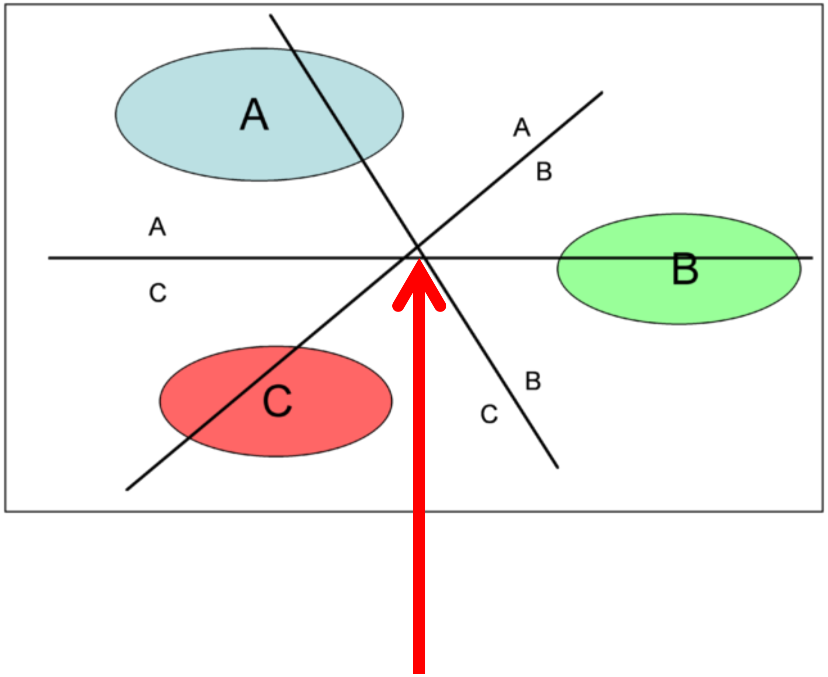


One-against-rest

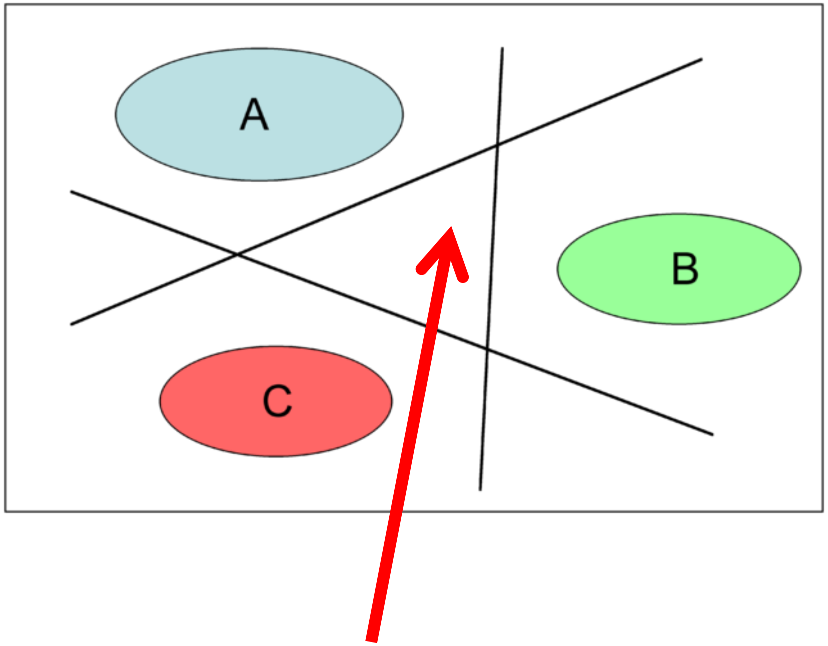


Problems

One-against-one



One-against-rest



Crammer & Singer's formulation

- A Naïve approach

$$\begin{aligned} \min_{w_1,\dots,w_K} \quad & \frac{1}{2}\|(w_1,\dots,w_K)\|^2 + C\sum_{ik}\xi_{ik} \\ \text{s.t.} \quad & \forall(i,k), \quad {w_{y^i}}^\top x^i - w_k^\top x^i \geq \mathbf{1}\{k \neq y^i\} - \xi_{ik} \end{aligned}$$

Crammer & Singer's formulation

- A Naïve approach

$$\min_{w_1,\ldots,w_K} \quad \frac{1}{2}\|(w_1,\ldots,w_K)\|^2 + C\sum_{ik}\xi_{ik}$$

$$\text{s.t.} \quad \forall(i,k), \quad {w_{y^i}}^\top x^i - w_k^\top x^i \geq \mathbf{1}\{k \neq y^i\} - \xi_{ik}$$

- C & S's formulation

$$\min_{\mathbf{w}_1,\ldots,\mathbf{w}_k,\xi} \quad \frac{1}{2}\beta\sum_{p=1}^{k}||\mathbf{w}_p||^2 + \sum_{i=1}^{n}\xi_i$$

$$s.t. \quad \forall i = 1,\ldots,n, r = 1,\ldots,k$$

$$\mathbf{w}_{y_i}^T\mathbf{x}_i + \delta_{y_i,r} - \mathbf{w}_r^T\mathbf{x}_i \geq 1 - \xi_i$$

Error-Correcting Output Code (ECOC)

Table 3: A 15-bit error-correcting output code for a ten-class problem.

Class	Code Word														
	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}
0	1	1	0	0	0	0	1	0	1	0	0	1	1	0	1
1	0	0	1	1	1	1	0	1	0	1	1	0	0	1	0
2	1	0	0	1	0	0	0	1	1	1	1	0	1	0	1
3	0	0	1	1	0	1	1	1	0	0	0	0	1	0	1
4	1	1	1	0	1	0	1	1	0	0	1	0	0	0	1
5	0	1	0	0	1	1	0	1	1	1	0	0	0	0	1
6	1	0	1	1	1	0	0	0	0	1	0	1	0	0	1
7	0	0	0	1	1	1	1	0	1	0	1	1	0	0	1
8	1	1	0	1	0	1	1	0	0	1	0	0	0	1	1
9	0	1	1	1	0	0	0	0	1	0	1	0	0	1	1

Codeword

Meta-classifier

0
1
0
0
0
0
0
0
0
0

Source: Dietterich and Bakiri (1995)

Discussions

D&B suggest that M be constructed to have good *error-correcting* properties — if the minimum Hamming distance between rows of M is d, then the resulting multiclass classification will be able to correct any $\lfloor \frac{d-1}{2} \rfloor$ errors.

In learning, good *column* separation is also important — two identical (or opposite) columns will make identical errors. This highlights the key difference between the multiclass machine learning framework and standard error-correcting code applications.

Special cases of ECOC

One-against-one

All-pairs → k by $\binom{k}{2}$

$$
\begin{matrix}
+1 & +1 & +1 & 0 & 0 & 0 \\
-1 & 0 & 0 & +1 & +1 & 0 \\
0 & -1 & 0 & -1 & 0 & +1 \\
0 & 0 & -1 & 0 & -1 & -1
\end{matrix}
$$

One-against-rest

One-against-all → k by k

$$
\begin{matrix}
+1 & -1 & -1 & -1 \\
-1 & +1 & -1 & -1 \\
-1 & -1 & +1 & -1 \\
-1 & -1 & -1 & +1
\end{matrix}
$$

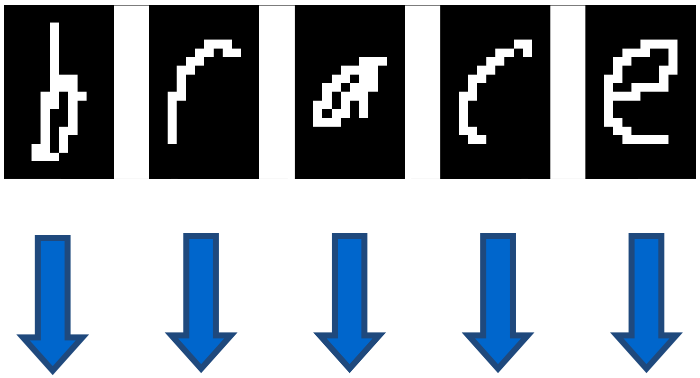
Empirical Study

- **In Defense of One-Vs-All Classification – *JMLR 2004***
  - The most important step in good multiclass classification is to **use the best binary classifier available**. Once this is done, it seems to make little difference what multiclass scheme is applied, and therefore a simple scheme such as OVA (or AVA) is preferable to a more complex error-correcting coding scheme or single-machine scheme.

Part II: Structured SVM

Slides Courtesy: Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, Yasemin Altun

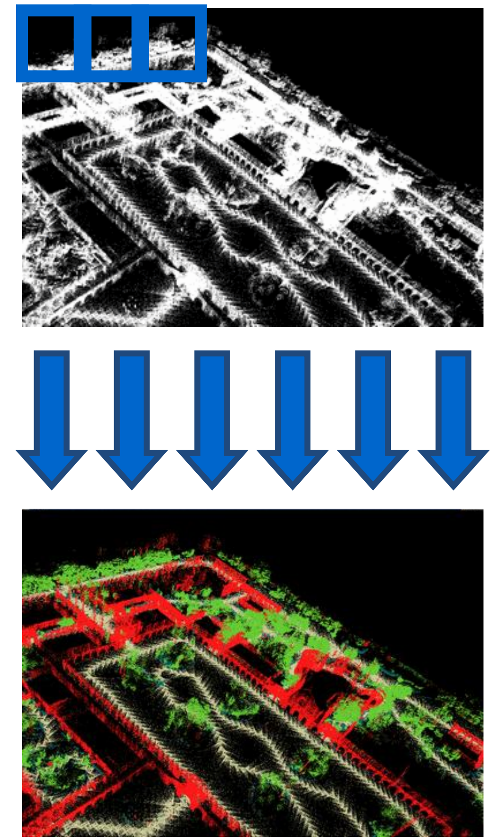
Local Classification



b r a r e

x

y

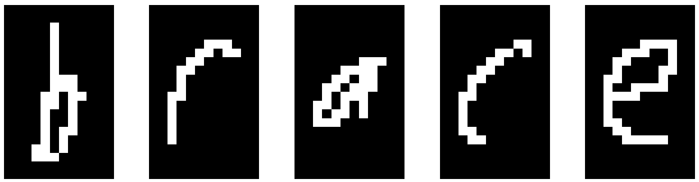


Classify using local information

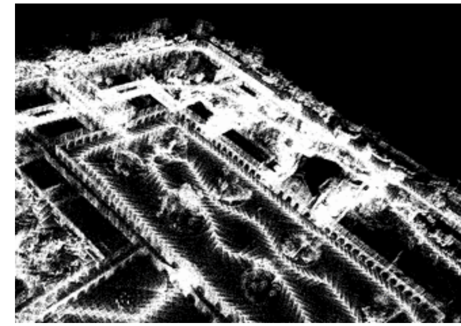
$\Rightarrow$ Ignores correlations!

[thanks to Ben Taskar for slide!]

Structured Classification

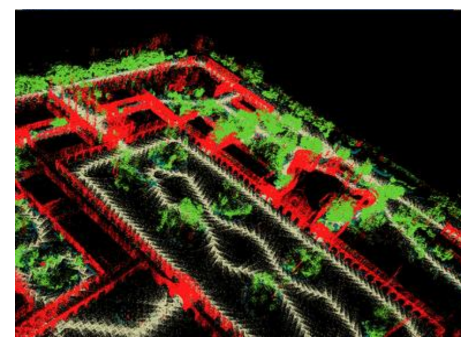


$\mathbf{x}$



b r a c e

$\mathbf{y}$



- Use local information
- Exploit correlations

[thanks to Ben Taskar for slide!]

Structured Prediction

- Given: labeled training data $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n) \in \mathcal{X} \times \mathcal{Y}$
- Task: learn mapping from inputs $\mathbf{x} \in \mathcal{X}$ to outputs $\mathbf{y} \in \mathcal{Y}$
- Special cases
  - Binary classification: $\mathcal{Y} = \{-1, 1\}$
  - Multiclass classification: $\mathcal{Y} = \{1, ..., k\}$
- Naive approach: treat each possible output in $\mathcal{Y}$ as discrete label
  - Big problem: $\mathcal{Y}$ gets huge

Exploiting Structure

- Enumerating all members of $\mathcal{Y}$ is often intractable
- Hence, try to exploit structure and dependencies within the output space
- Approach: learn discriminant function $F : \mathcal{X} \times \mathcal{Y} \rightarrow \Re$
- Hypotheses have the form: $f(\mathbf{x}; \mathbf{w}) = \underset{\mathbf{y} \in \mathcal{Y}}{\operatorname{argmax}} F(\mathbf{x}, \mathbf{y}; \mathbf{w})$
  - Parametrized by parameters $\mathbf{w}$
- Assume F linear in combined feature representation $\Psi(\mathbf{x}, \mathbf{y})$

$$F(\mathbf{x}, \mathbf{y}; \mathbf{w}) = \langle \mathbf{w}, \Psi(\mathbf{x}, \mathbf{y}) \rangle$$

Optimization Problem

- First, assume separable data, i.e.,

$$\forall i,\ \forall \mathbf{y} \in \mathcal{Y} \setminus \mathbf{y}_i :\ \langle \mathbf{w}, \delta\Psi_i(\mathbf{y}) \rangle > 0\,, \quad \delta\Psi_i(\mathbf{y}) \equiv \Psi(\mathbf{x}_i, \mathbf{y}_i) - \Psi(\mathbf{x}_i, \mathbf{y})$$

- Requiring that margin ≥ 1 gives the optimization problem

$$\mathrm{SVM}_0 : \min_{\mathbf{w}} \ \frac{1}{2}\|\mathbf{w}\|^2$$
$$\forall i,\ \forall \mathbf{y} \in \mathcal{Y} \setminus \mathbf{y}_i :\ \langle \mathbf{w}, \delta\Psi_i(\mathbf{y}) \rangle \geq 1\,.$$

- To handle non-separable data, add one slack variable per point

$$\mathrm{SVM}_1 : \min_{\mathbf{w}, \boldsymbol{\xi}} \ \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n} \xi_i, \ \text{ s.t. } \forall i, \xi_i \geq 0$$
$$\forall i,\ \forall \mathbf{y} \in \mathcal{Y} \setminus \mathbf{y}_i :\ \langle \mathbf{w}, \delta\Psi_i(\mathbf{y}) \rangle \geq 1 - \xi_i\,.$$

Replacing the Zero-one Loss

$$\mathrm{SVM}_1 : \min_{\mathbf{w}, \boldsymbol{\xi}} \ \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i, \ \text{ s.t. } \forall i, \xi_i \geq 0$$

$$\forall i, \forall \mathbf{y} \in \mathcal{Y} \setminus \mathbf{y}_i : \ \langle \mathbf{w}, \delta\Psi_i(\mathbf{y}) \rangle \geq 1 - \xi_i \, .$$

- Problem: all margin violations are penalized identically
- A better way: assume existence of loss function $\triangle(\mathrm{y}, \hat{\mathrm{y}})$ to model closeness of predictions
- First idea: scale slack variables by loss incurred

$$\mathrm{SVM}_1^{\triangle s} : \min_{\mathbf{w}, \boldsymbol{\xi}} \ \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i, \ \text{ s.t. } \forall i, \xi_i \geq 0$$

$$\forall i, \forall \mathbf{y} \in \mathcal{Y} \setminus \mathbf{y}_i : \langle \mathbf{w}, \delta\Psi_i(\mathbf{y}) \rangle \geq 1 - \frac{\xi_i}{\triangle(\mathbf{y}_i, \mathbf{y})} .$$

Replacing the Zero-one Loss

$$\mathrm{SVM}_1: \min_{\mathbf{w}, \boldsymbol{\xi}} \ \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i, \ \text{ s.t. } \forall i, \xi_i \geq 0$$

$$\forall i, \ \forall \mathbf{y} \in \mathcal{Y} \setminus \mathbf{y}_i: \ \langle \mathbf{w}, \delta\Psi_i(\mathbf{y})\rangle \geq 1 - \xi_i \, .$$

- Problem: all margin violations are penalized identically
- A better way: assume existence of loss function $\triangle(\mathrm{y}, \hat{\mathrm{y}})$ to model closeness of predictions
- Another idea: scale the margin by the loss (e.g., Hamming loss in Max-Margin Markov Networks):

$$\forall i, \ \forall \mathbf{y} \in \mathcal{Y} \setminus \mathbf{y}_i: \ \langle \mathbf{w}, \delta\Psi_i(\mathbf{y})\rangle \geq \triangle(\mathbf{y}_i, \mathbf{y}) - \xi_i$$

  - Also yields bound on empirical risk

Dual Optimization Problem

- Write dual problem using SVM-like Lagrangian techniques
  - (take Lagrangian, find saddle point with KKT conditions, plug back into objective function)
  - For SVM_0, we get $\max_{\alpha} \sum_{i,\mathbf{y}\neq\mathbf{y}_i} \alpha_{i\mathbf{y}} - \frac{1}{2} \sum_{\substack{i,\mathbf{y}\neq\mathbf{y}_i \\ j,\bar{\mathbf{y}}\neq\mathbf{y}_j}} \alpha_{i\mathbf{y}} \alpha_{j\bar{\mathbf{y}}} \left\langle \delta\Psi_i(\mathbf{y}), \delta\Psi_j(\bar{\mathbf{y}}) \right\rangle$

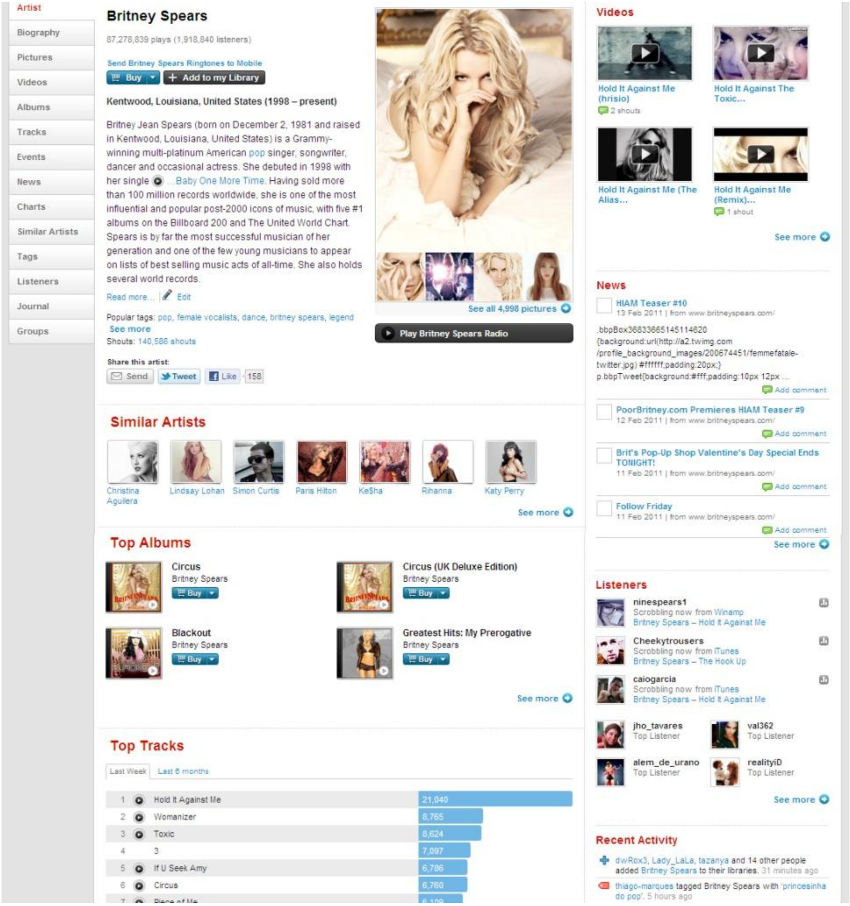
    $$\text{s.t. } \forall i, \forall \mathbf{y} \neq \mathcal{Y} \setminus \mathbf{y}_i : \quad \alpha_{i\mathbf{y}} \geq 0 \,.$$
  - As in SVMs, can replace inner product with kernel:

    $$K((\mathbf{x}, \mathbf{y}), (\mathbf{x}', \mathbf{y}'))$$
  - Other optimization problems are also easily formulated in terms of dual QPs

Case Study: Max Margin Learning on Domain-Independent Web Information Extraction

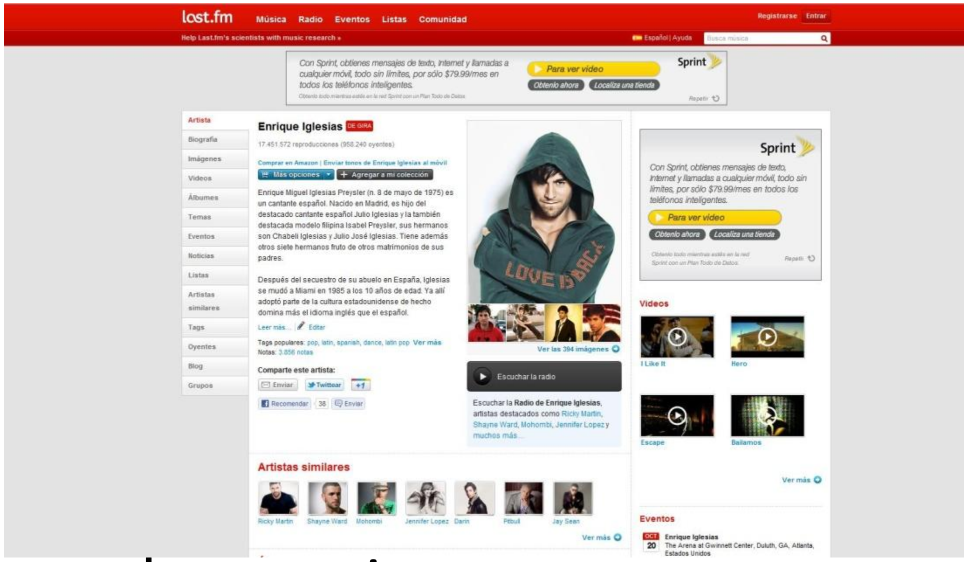
Motivation

- "Understand" web page
  - Assign semantics to each component of a page
  - Understand functionality of each section
  - Distinguish main content from side contents
- Page layout reveals important cues



Motivation (Cont.)

- Human can “understand” a page written in foreign language



  - Position
  - Size
  - Font
  - Color
  - Boldness
  - Relative position to other sections
- Can a computer fulfill similar task?
  - Domain-independent web information extraction

The Proposed Approach

- Page Segmentation: vision tree
  - Built based on DOM tree
  - With visual information
  - Correspond to how each node is displayed
- Structured Segmentation
  - Assign label for each node in the vision tree
  - Information extraction based on node classification

Structured Classification

- Label space for **leaf** nodes
  - *attribute name, attribute value, non attribute, image, nav bar, main title, page tail, image caption*
  - *non-atribute (anything else)*
- Label space for **non-leaf** nodes
  - *structure block, data record, nav bar block, non attribute block, value block, page tail block, name block, image block, image caption block, main title block*

Max Margin Learning

- Structured output space (tree)
  - If treated as conventional classification: exponential number of classes → unable to train
  - Augmented loss: misclassify a tree by one node should receive much less penalty than misclassifying the entire tree
- Input-output feature mapping: $\Phi(\mathbf{x}_l, \mathbf{y}_l) : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{R}^d$
- Linear discriminant function: $F(\mathbf{x}, \mathbf{y}) = \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$
- Response to input x:

$$h(x) = \arg\max_{\mathbf{y} \in \mathcal{Y}} F(\mathbf{x}, \mathbf{y})$$

Max Margin Learning (Cont.)

- Augmented loss: $\Delta(\mathbf{y}, \mathbf{y}')$
  - # of disagreements between y and y'
- Learning: use SVM to find optimal w

$$\min_{\mathbf{w}} \quad \frac{1}{2}||\mathbf{w}||^2 + C\sum_{l=1}^{m} \xi_l$$

$$s.t. \quad \mathbf{w}^T\Phi(\mathbf{x}_l, \mathbf{y}_l) - \mathbf{w}^T\Phi(\mathbf{x}_l, \mathbf{y}') \geq \Delta(\mathbf{y}_l, \mathbf{y}') - \xi_l, \forall \mathbf{y}', \forall l$$

$$\xi_l \geq 0, \ \forall l$$

- Inference: dynamical programming

$$\mathbf{y}^* = \arg\max_{\mathbf{y}\in\mathcal{Y}} F(\mathbf{x}, \mathbf{y}) = \arg\max_{\mathbf{y}\in\mathcal{Y}} \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$$

Input-Output Feature Mapping

- Two types of cliques in the hierarchical model
  - Cliques $\mathcal{C}^b$ covering observation-state node pairs
  - Cliques $\mathcal{C}^p$ covering state-state node pairs
- Correspondingly, two types of features
  - Features describing cliques $\mathcal{C}^b$

$$\phi^b_{ld}(\mathbf{x}, \mathbf{y}) = \sum_{(\mathbf{x}_i, y_i) \in \mathcal{C}^b} \mathbb{I}_{[y_i = l]} \phi(\mathbf{x}_i)_d$$

  - Features describing cliques $\mathcal{C}^p$

$$\phi^p_{l\bar{l}}(\mathbf{x}, \mathbf{y}) = \sum_{(y_i, y_j) \in \mathcal{C}^p} \mathbb{I}_{[y_i = l]} \mathbb{I}_{[y_j = \bar{l}]}$$

Input-Output Feature Mapping: Type I

- Spatial features
  - Position of block center, block height, block weight, block area
- Features for all nodes
  - Link number, number of various HTML tags, number of child nodes
- Features for leaf nodes only
  - Text length, font, bold, italic, word count, number of images, image size, link text length

Input-Output Feature Mapping: Type II

- Parent-child relationship
  - Label co-occurrence pattern

$$\forall l, \bar{l}: \quad \phi^{p}_{st}(l, \bar{l}) = \mathbb{I}_{[y_s = l]} \mathbb{I}_{[y_t = \bar{l}]}$$

- Connect spatially adjacent blocks
  - Link a node with its k nearest neighbors
  - Define label co-occurrence pattern for connected node pair *(i,j)*

$$\forall l, \bar{l}: \quad \phi^{s}_{ij}(l, \bar{l}) = \mathbb{I}_{[y_i = l]} \mathbb{I}_{[y_j = \bar{l}]}$$

Learning and Inference

- Learning
  - Quadratic programming with exponential number of constraints → cutting plane algorithm (a.k.a., constraint generation, bundle method)
- Inference
  - Without edges between spatially adjacent blocks → dynamical programming (a.k.a. Viterbi decoding)
  - With edges between spatially adjacent blocks → loopy belief propagation

Empirical Study

- Block level prediction results

	Multiclass SVM	SVM-Tree	SVM-Cyclic
Accuracy	52.11	71.93	**74.96**
pName	65.55	69.08	**72.62**
rName	12.13	69.83	**73.41**
pValue	52.55	70.69	**74.00**
rValue	35.78	75.41	**78.55**

- Attribute name-value pair extraction
  - 1000 web pages
  - Precision: 56.91%
  - Recall: 59.37%

Thank you