Decision-theoretic Approach to Multi-label Classification

Krzysztof Dembczyński
Intelligent Decision Support Systems Laboratory (IDSS)
Poznań University of Technology, Poland

TFML, Bedlewo, Poland, 20.02.2015
1 Multi-label classification
2 Simple approaches to multi-label classification
3 Beyond simple approaches
4 Maximization of the F-measure
5 Rank loss minimization
6 Summary
1 Multi-label classification

2 Simple approaches to multi-label classification

3 Beyond simple approaches

4 Maximization of the F-measure

5 Rank loss minimization

6 Summary
Image annotation: cloud? sky? tree?
Ecology: Prediction of the presence or absence of species
Stack Overflow is a question and answer site for professional and enthusiast programmers. It's 100% free, no registration required.

Take the 2-minute tour

Top Questions

<table>
<thead>
<tr>
<th>Votes</th>
<th>Answers</th>
<th>Views</th>
<th>Title</th>
<th>Tags</th>
<th>Votes</th>
<th>Answers</th>
<th>Views</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>sending and receiving mails from registered user emailaddresses</td>
<td>php, email, web-applications</td>
<td>411</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>5</td>
<td>How to create sprites using ConfigParser in Pygame</td>
<td>python, pygame</td>
<td>411</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>8</td>
<td>Fortran: possible fibonacci logical error</td>
<td>fortran, fibonacci, fortran95</td>
<td>411</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>1k</td>
<td>Angular - Using one controller for many coherent views across multiple HTTP requests</td>
<td></td>
<td>411</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Document tagging
Multi-label classification

- **Multi-label classification**: For a feature vector $\mathbf{x}$ predict accurately a vector of binary responses $\mathbf{y}$ using a function $h(\mathbf{x})$:

$$
\mathbf{x} = (x_1, x_2, \ldots, x_p) \xrightarrow{h(\mathbf{x})} \mathbf{y} = (y_1, y_2, \ldots, y_m) \in \mathcal{Y} = \{0, 1\}^m
$$
Multi-label classification

- Training data: \( \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \).
- **Predict** a vector \( y = (y_1, y_2, \ldots, y_m) \) for a given \( x \).

<table>
<thead>
<tr>
<th></th>
<th>( x_1 )</th>
<th>( x_2 )</th>
<th>( y_1 )</th>
<th>( y_2 )</th>
<th>\ldots</th>
<th>( y_m )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>5.0</td>
<td>4.5</td>
<td>1</td>
<td>1</td>
<td>\ldots</td>
<td>0</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>2.0</td>
<td>2.5</td>
<td>0</td>
<td>1</td>
<td>\ldots</td>
<td>0</td>
</tr>
<tr>
<td>( x_n )</td>
<td>3.0</td>
<td>3.5</td>
<td>0</td>
<td>1</td>
<td>\ldots</td>
<td>1</td>
</tr>
</tbody>
</table>

| \( x \) | 4.0      | 2.5      | \?       | \?       | \?      | ?        |
Multi-label classification

- Training data: \( \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \).
- **Predict** a vector \( y = (y_1, y_2, \ldots, y_m) \) for a given \( x \).

<table>
<thead>
<tr>
<th></th>
<th>( x_1 )</th>
<th>( x_2 )</th>
<th>( y_1 )</th>
<th>( y_2 )</th>
<th>\ldots</th>
<th>( y_m )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>( x_1 )</strong></td>
<td>5.0</td>
<td>4.5</td>
<td>1</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td><strong>( x_2 )</strong></td>
<td>2.0</td>
<td>2.5</td>
<td>0</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td><strong>( x_n )</strong></td>
<td>3.0</td>
<td>3.5</td>
<td>0</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td><strong>( x )</strong></td>
<td>4.0</td>
<td>2.5</td>
<td>1</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
Supervised learning

Training data \( \{ x_i, y_i \} \)

Learning algorithm

Model \( f(x, y) \)

Prediction by using \( h(x) \)

Test example \( x \)

Predicted outcome \( \hat{y} = h(x) \)

True outcome \( y \)

Evaluation

Risk

Estimated risk \( \approx \) Loss \( \ell(y, \hat{y}) \)
Supervised learning

Training data: \( \{ x_i, y_i \} \)

Learning algorithm: Model \( f(x, y) \)

Prediction: \( \hat{y} = h(x) \)

Test example: \( x \)

Predicted outcome: \( \hat{y} \)

True outcome: \( y \)

Evaluation: Risk

Estimated risk: \( \approx \text{Loss} \( y, \hat{y} \) \)
Supervised learning

Training data: \( \{x_i, y_i\}_{i=1}^n \)

Learning algorithm: produces a model \( f(x, y) \)

Prediction by using \( h(x) \)

Test example \( x \)

Predicted outcome: \( \hat{y} = h(x) \)

True outcome: \( y \)

Evaluation:

Risk: \( \approx \) Loss \( \ell(y, \hat{y}) \)
Supervised learning

Training data: 
\{x_i, y_i\}_{n=1}^{n}

Learning algorithm:
Model: \( f(x, y) \)

Prediction by using: 
\( h(x) \)

Test example: 
\( x \)

Predicted outcome:
\( \hat{y} = h(x) \)

Evaluation:
Risk:
Estimated risk: 
\( \approx \) Loss: 
\( \ell(y, \hat{y}) \)
Supervised learning

Training data \( \{x_i, y_i\} \)

Learning algorithm

Model \( f(x, y) \)

Prediction by using \( h(x) \)

Test example \( x \)

Predicted outcome \( \hat{y} = h(x) \)

True outcome \( y \)

Evaluation

Risk

Estimated risk \( \approx \) Loss \( \ell(y, \hat{y}) \)
Supervised learning

- Training data: \(\{x_i, y_i\}_{i=1}^n\)
- Learning algorithm: Model \(f(x, y)\)
- Prediction: \(\hat{y} = h(x)\)
- Test example \(x\)
- True outcome \(y\)
- Predicted outcome \(\hat{y}\)
- Evaluation Risk: \(\ell(y, \hat{y})\)
- Estimated risk: \(\approx \text{Loss } \ell(y, \hat{y})\)
Supervised learning

Training data \{ x_i, y_i \}_{i=1}^n

Learning algorithm

Model \( f(x, y) \)

Prediction by using \( h(x) \)

Test example \( x \)

Predicted outcome \( \hat{y} = h(x) \)

True outcome \( y \)

Loss \( \ell(y, \hat{y}) \)

Risk

Evaluation

Estimated risk \( \approx \) Loss \( \ell(y, \hat{y}) \)
Supervised learning

Training data
\[[x_i, y_i]_{i=1}^{n}\]

Learning algorithm
Model
\[f(x, y)\]

Prediction by using \(h(x)\)

Test example \(x\)

Predicted outcome \(\hat{y} = h(x)\)

True outcome \(y\)

Evaluation

Risk

Loss \(\ell(y, \hat{y})\)
Supervised learning

Training data \( \{x_i, y_i\}_{1}^{n} \)

Learning algorithm

Test example \( x \)

Prediction by using \( h(x) \)

Predicted outcome \( \hat{y} = h(x) \)

True outcome \( y \)

Loss \( \ell(y, \hat{y}) \)

Risk

Evaluation
Supervised learning

Training data \( \{x_i, y_i\}_{i=1}^n \)

Learning algorithm

Model \( f(x, y) \)

Prediction by using \( h(x) \)

Test example \( x \)

Loss \( \ell(y, \hat{y}) \)

Risk

Evaluation

Estimated risk \( \approx \) Loss \( \ell(y, \hat{y}) \)

Predicted outcome \( \hat{y} = h(x) \)

True outcome \( y \)
Supervised learning

Training data \( \{x_i, y_i\}_{i=1}^n \)

Learning algorithm

Model \( f(x, y) \)

Loss \( \ell(y, \hat{y}) \)

Test example \( x \)

Prediction by using \( h(x) \)

Predicted outcome \( \hat{y} = h(x) \)

True outcome \( y \)

Risk
Supervised learning

Training data \( \{x_i, y_i\}_{i=1}^{n} \)

Learning algorithm

Model \( f(x, y) \)

Prediction by using \( h(x) \)

Test example \( x \)

Prediction \( \hat{y} = h(x) \)

True outcome \( y \)

Loss \( \ell(y, \hat{y}) \)

Risk

Estimated risk

Evaluation

Risk

Estimated risk

Evaluation
Supervised learning

Training data \( \{x_i, y_i\}_{1}^{n} \)

Learning algorithm

Model \( f(x, y) \)

Evaluation

Loss \( \ell(y, \hat{y}) \)

Risk

Estimated risk

Prediction by using \( h(x) \)

Predicted outcome \( \hat{y} = h(x) \)

Evaluation

Test example \( x \)

True outcome \( y \)
Supervised learning

Training data \( \{x_i, y_i\}_{i=1}^n \)

Learning algorithm

Prediction by using \( h(x) \)

Test example \( x \)

Model \( f(x, y) \)

Loss \( \ell(y, \hat{y}) \)

Risk

Estimated risk

Evaluation

True outcome \( y \)

Predicted outcome \( \hat{y} = h(x) \)

Evaluation

\( \approx \)
Multi-label classification

- **Example** $x$ is coming from an unknown input distribution $P(x)$.
- **True outcome** $y$ is generated from $P(y \mid x)$.
- **Predicted outcome** is given by $\hat{y} = h(x)$.
- The (task) **loss** of a single prediction is $\ell(y, \hat{y})$. 
Multi-label classification

• The overall goal is to minimize the risk:

\[ L_\ell(h) = \mathbb{E}_{(x,y)}(\ell(y, h(x))) \]

• The optimal prediction function, the so-called Bayes classifier, is:

\[ h^*_\ell = \arg \min_h L_\ell(h) \]

• The regret of a classifier \( h \) with respect to \( \ell \) is defined as:

\[ \text{Reg}_\ell(h) = L_\ell(h) - L_\ell(h^*_\ell) = L_\ell(h) - L^*_\ell \]
Multi-label classification

• We use training examples \( \{x_i, y_i\}_{i=1}^n \) to find either:
  ▶ a good approximation of \( h^* \), or
  ▶ a good estimation of \( P(y|x) \) (or a function of it).

• In the second case, we need to apply an inference procedure to approximate \( h^* \).
Main challenges

• Appropriate modeling of dependencies between labels $y_1, y_2, \ldots, y_m$

• A multitude of multivariate loss functions defined over the output vector $\ell(y, h(x))$
Marginal and conditional dependence:

$marginal \ (in)dependence \neq conditional \ (in)dependence$

$$P(y) \neq \prod_{i=1}^{m} P(y_i) \quad P(y \mid x) \neq \prod_{i=1}^{m} P(y_i \mid x)$$

Structure imposed (domain knowledge) on labels:

- Chains,
- Hierarchies,
- General graphs,
- ...
Multi-label loss functions

- **Decomposable** and non-decomposable losses over labels

\[ \ell(y, h(x)) = \sum_{i=1}^{m} \ell(y_i, h_i(x)) \quad \ell(y, h(x)) \neq \sum_{i=1}^{m} \ell(y_i, h_i(x)) \]

- Different formulations of loss functions possible:
  - Set-based losses.
  - Ranking-based losses.
Multi-label loss functions

- **Subset 0/1 loss**: \( \ell_{0/1}(y, h) = [y \neq h] \)

- **Hamming loss**: \( \ell_H(y, h) = \frac{1}{m} \sum_{i=1}^{m} [y_i \neq h_i] \)

- **F-measure-based loss**: \( \ell_F(y, h) = 1 - \frac{2 \sum_{i=1}^{m} y_i h_i}{\sum_{i=1}^{m} y_i + \sum_{i=1}^{m} h_i} \)

- **Rank loss**: \( \ell_{\text{rnk}}(y, f) = w(y) \sum_{y_i > y_j} \left( [f_i < f_j] + \frac{1}{2} [f_i = f_j] \right) \)

- ...
The set-based loss function \( \ell(y, h) \) should fulfill some basic conditions:

- \( \ell(y, h) = 0 \) if and only if \( y = h \).
- \( \ell(y, h) \) is maximal when \( y_i \neq h_i \) for every \( i = 1, \ldots, m \).
- Should be monotonically non-decreasing with respect to the number of \( y_i \neq h_i \).
Relations between losses

- The set-based loss function $\ell(y, h)$ should fulfill some basic conditions:
  - $\ell(y, h) = 0$ if and only if $y = h$.
  - $\ell(y, h)$ is maximal when $y_i \neq h_i$ for every $i = 1, \ldots, m$.
  - Should be monotonically non-decreasing with respect to the number of $y_i \neq h_i$.

- In case of deterministic data (no-noise): the optimal prediction should have the same form for all loss functions and the risk for this prediction should be 0.
Relations between losses

• The set-based loss function $\ell(y, h)$ should fulfill some basic conditions:
  - $\ell(y, h) = 0$ if and only if $y = h$.
  - $\ell(y, h)$ is maximal when $y_i \neq h_i$ for every $i = 1, \ldots, m$.
  - Should be monotonically non-decreasing with respect to the number of $y_i \neq h_i$.

• **In case of deterministic data (no-noise):** the optimal prediction should have the same form for all loss functions and the risk for this prediction should be 0.

• **In case of non-deterministic data (noise):** the optimal prediction and its risk can be different for different losses.
The loss functions, like Hamming loss or subset 0/1 loss, often referred to as task losses, are usually neither convex nor differentiable.
Learning and inference with multi-label losses

- The loss functions, like Hamming loss or subset 0/1 loss, often referred to as **task losses**, are usually neither convex nor differentiable.
- Therefore learning is a hard optimization problem.
Learning and inference with multi-label losses

- The loss functions, like Hamming loss or subset 0/1 loss, often referred to as **task losses**, are usually neither convex nor differentiable.
- Therefore learning is a hard optimization problem.
- Two approaches try to make this task easier
  - Reduction.
  - Surrogate loss minimization.
• The loss functions, like Hamming loss or subset 0/1 loss, often referred to as task losses, are usually neither convex nor differentiable.
• Therefore learning is a hard optimization problem.
• Two approaches try to make this task easier
  ▶ Reduction.
  ▶ Surrogate loss minimization.
• Two phases in solving multi-label problems:
  ▶ Learning: Estimate parameters of a scoring function $f(x, y)$.
  ▶ Inference: Use the scoring function $f(x, y)$ to classify new instances by finding the best $y$ for a given $x$. 
Reduction

- **Reduce** the original problem into simple problems, for which efficient algorithmic solutions are available.
- Reduction to one or a sequence of problems.
- Plug-in rule classifiers.

\[
{\{(x, y)\}_{i=1}^{n}} \\
\downarrow \\
(x, y) \rightarrow (x', y') \\
\downarrow\downarrow\downarrow\downarrow\downarrow \\
\text{LEARNING} \\
\min \tilde{\ell}(y', x', f) \\
\downarrow\downarrow\downarrow\downarrow\downarrow \\
f(x', y') \\
\downarrow \\
x \rightarrow \text{Inference} \rightarrow \hat{y}
\]
Surrogate loss minimization

\[ \{(x, y)\}_{i=1}^n \]

- Replace the task loss by a **surrogate loss** that is easier to cope with.
- Surrogate loss is typically a differentiable approximation of the task loss or a convex upper bound of it.
Statistical consistency

- Analysis of algorithms in terms of their infinite sample performance.\(^1\)
- We say that a proxy loss \( \tilde{\ell} \) is consistent (calibrated) with the task loss \( \ell \) when the following holds:

\[
\text{Reg}_{\tilde{\ell}}(h) \to 0 \Rightarrow \text{Reg}_{\ell}(h) \to 0.
\]

- The definition concerns both surrogate loss minimization and reduction:
  - Surrogate loss minimization: \( \tilde{\ell} = \) surrogate loss.
  - Reduction: \( \tilde{\ell} = \) loss used in the reduced problem.

---


1 Multi-label classification

2 Simple approaches to multi-label classification

3 Beyond simple approaches

4 Maximization of the F-measure

5 Rank loss minimization

6 Summary
Basic reductions: Binary relevance

- **Binary relevance**: Decomposes the problem to $m$ binary classification problems:

$$(x, y) \mapsto (x, y = y_i), \quad i = 1, \ldots, m$$

<table>
<thead>
<tr>
<th></th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$y_1$</th>
<th>$y_2$</th>
<th>\ldots</th>
<th>$y_m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>5.0</td>
<td>4.5</td>
<td>1</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>$x_2$</td>
<td>2.0</td>
<td>2.5</td>
<td>0</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>\vdots</td>
<td>\vdots</td>
<td>\vdots</td>
<td>\vdots</td>
<td>\vdots</td>
<td>\vdots</td>
<td>\vdots</td>
</tr>
<tr>
<td>$x_n$</td>
<td>3.0</td>
<td>3.5</td>
<td>0</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>
Basic reductions: Binary relevance

- **Binary relevance**: Decomposes the problem to \( m \) binary classification problems:

\[
(x, y) \rightarrow (x, y_i), \quad i = 1, \ldots, m
\]

<table>
<thead>
<tr>
<th>( x )</th>
<th>( y )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>5.0</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>2.0</td>
</tr>
<tr>
<td>( \vdots )</td>
<td>( \vdots )</td>
</tr>
<tr>
<td>( x_n )</td>
<td>3.0</td>
</tr>
</tbody>
</table>

- Seems to be very simplistic.
- Ignores any dependencies.
- Is it good for any loss function?
Basic reductions: Label powerset

- **Label powerset**: Treats each label combination as a new meta-class in multi-class classification:

\[(x, y) \rightarrow (x, y = \text{metaclass}(y))\]

<table>
<thead>
<tr>
<th></th>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(y_1)</th>
<th>(y_2)</th>
<th>(\cdots)</th>
<th>(y_m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x_1)</td>
<td>5.0</td>
<td>4.5</td>
<td>1</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>(x_2)</td>
<td>2.0</td>
<td>2.5</td>
<td>0</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>(\vdots)</td>
<td>(\vdots)</td>
<td>(\vdots)</td>
<td>(\vdots)</td>
<td>(\vdots)</td>
<td></td>
<td>(\vdots)</td>
</tr>
<tr>
<td>(x_n)</td>
<td>3.0</td>
<td>3.5</td>
<td>0</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>
Basic reductions: Label powerset

- **Label powerset**: Treats each label combination as a new meta-class in multi-class classification:

\[(x, y) \mapsto (x, y = \text{metaclass}(y))\]

<table>
<thead>
<tr>
<th></th>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(y_1)</th>
<th>(y_2)</th>
<th>(\cdots)</th>
<th>(y_m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x_1)</td>
<td>5.0</td>
<td>4.5</td>
<td>1</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>(x_2)</td>
<td>2.0</td>
<td>2.5</td>
<td>0</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>(x_n)</td>
<td>3.0</td>
<td>3.5</td>
<td>0</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

- Any multi-class classification algorithm can be used, but the number of classes is huge.
- Takes other labels into account, but ignores internal structure of classes (label vectors).
What about task losses minimized by BR and LP?
Synthetic data

- Two independent models:
  \[ f_1(x) = \frac{1}{2} x_1 + \frac{1}{2} x_2, \quad f_2(x) = \frac{1}{2} x_1 - \frac{1}{2} x_2 \]

- Logistic model to get labels:
  \[ P(y_i = 1) = \frac{1}{1 + \exp(-2f_i)} \]
Synthetic data

- Two dependent models:
  \[ f_1(x) = \frac{1}{2}x_1 + \frac{1}{2}x_2 \quad f_2(y_1, x) = y_1 + \frac{1}{2}x_1 - \frac{1}{2}x_2 - \frac{2}{3} \]

- Logistic model to get labels:
  \[ P(y_i = 1) = \frac{1}{1 + \exp(-2f_i)} \]
Results for two performance measures

- **Hamming loss:** \( \ell_H(y, h) = \frac{1}{m} \sum_{i=1}^{m} [y_i \neq h_i] \).
- **Subset 0/1 loss:** \( \ell_{0/1}(y, h) = [y \neq h] \).

<table>
<thead>
<tr>
<th>Conditional independence</th>
<th>Conditional dependence</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Classifier</strong></td>
<td><strong>Hamming loss</strong></td>
</tr>
<tr>
<td>BR LR</td>
<td></td>
</tr>
<tr>
<td>LP LR</td>
<td></td>
</tr>
</tbody>
</table>
Results for two performance measures

- Hamming loss: \( \ell_H(y, h) = \frac{1}{m} \sum_{i=1}^{m} [y_i \neq h_i] \),

- Subset 0/1 loss: \( \ell_{0/1}(y, h) = [y \neq h] \).

<table>
<thead>
<tr>
<th>Conditional independence</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>classifier</td>
<td>Hamming loss</td>
<td>subset 0/1 loss</td>
</tr>
<tr>
<td>BR LR</td>
<td>0.4232</td>
<td>0.6723</td>
</tr>
<tr>
<td>LP LR</td>
<td>0.4232</td>
<td>0.6725</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Conditional dependence</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>classifier</td>
<td>Hamming loss</td>
<td>subset 0/1 loss</td>
</tr>
<tr>
<td>BR LR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LP LR</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Results for two performance measures

- Hamming loss: \( \ell_H(y, h) = \frac{1}{m} \sum_{i=1}^{m} [y_i \neq h_i] \),

- Subset 0/1 loss: \( \ell_{0/1}(y, h) = [y \neq h] \).

<table>
<thead>
<tr>
<th>Conditional independence</th>
<th>classifier</th>
<th>Hamming loss</th>
<th>subset 0/1 loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>BR LR</td>
<td>0.4232</td>
<td>0.6723</td>
<td></td>
</tr>
<tr>
<td>LP LR</td>
<td>0.4232</td>
<td>0.6725</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Conditional dependence</th>
<th>classifier</th>
<th>Hamming loss</th>
<th>subset 0/1 loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>BR LR</td>
<td>0.3470</td>
<td>0.5499</td>
<td></td>
</tr>
<tr>
<td>LP LR</td>
<td>0.3610</td>
<td>0.5146</td>
<td></td>
</tr>
</tbody>
</table>
Linear + XOR synthetic data

Figure: Problem with two targets: shapes ($\triangle$ vs. $\circ$) and colors ($\square$ vs. $\blacksquare$).
Linear + XOR synthetic data

<table>
<thead>
<tr>
<th>CLASSIFIER</th>
<th>HAMMING LOSS</th>
<th>SUBSET 0/1 LOSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>BR LR</td>
<td>0.2399(±0.0097)</td>
<td>0.4751(±0.0196)</td>
</tr>
<tr>
<td>LP LR</td>
<td>0.0143(±0.0020)</td>
<td>0.0195(±0.0011)</td>
</tr>
<tr>
<td>BAYES OPTIMAL</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
### Linear + XOR synthetic data

<table>
<thead>
<tr>
<th>CLASSIFIER</th>
<th>Hamming</th>
<th>Subset 0/1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LOSS</td>
<td>LOSS</td>
</tr>
<tr>
<td>BR LR</td>
<td>0.2399 (±0.0097)</td>
<td>0.4751 (±0.0196)</td>
</tr>
<tr>
<td>LP LR</td>
<td>0.0143 (±0.0020)</td>
<td>0.0195 (±0.0011)</td>
</tr>
<tr>
<td><strong>BR MLRules</strong></td>
<td><strong>0.0011 (±0.0002)</strong></td>
<td><strong>0.0020 (±0.0003)</strong></td>
</tr>
<tr>
<td><strong>Bayes Optimal</strong></td>
<td><strong>0</strong></td>
<td><strong>0</strong></td>
</tr>
</tbody>
</table>
• BR LR uses two linear classifiers: cannot handle the label color (□ vs. ■) – the XOR problem.
• LP LR uses four linear classifiers to solve 4-class problem (△, ▲, ○, ●): extends the hypothesis space.
• BR MLRules uses two non-linear classifiers (based on decision rules): XOR problem is not a problem.
• There is no noise in the data.
• Easy to perform unfair comparison.
Multi-label loss functions

• The conditional risk in multi-label classification of $h$ at $x$:

$$L_{\ell}(h \mid x) = \mathbb{E}_y [\ell(y, h(x))] = \sum_{y \in \mathcal{Y}} P(y \mid x) \ell(y, h(x))$$

• The risk-minimizing classifier for a given $x$:

$$h^*(x) = \arg \min_h L_{\ell}(h \mid x)$$

• Let us start with Hamming loss and subset 0/1 loss ...²

Hamming loss vs. subset 0/1 loss

- The risk minimizer for the Hamming loss is

  $h^*_{\text{marg}}(x) = \arg\max_{y_i \in \{0, 1\}} P(y_i | x)$, $i = 1, \ldots, m$.

- The risk minimizer for the subset 0/1 loss is

  $h^*_{\text{joint}}(x) = \arg\max_{y \in Y} P(y | x)$.

Marginal mode vs. joint mode.
Hamming loss vs. subset 0/1 loss

- The risk minimizer for the Hamming loss is the **marginal mode**:

$$h^*_i(x) = \arg \max_{y_i \in \{0,1\}} P(y_i | x), \quad i = 1, \ldots, m,$$

Marginal mode: 1 1 1 1

Joint mode: 0 0 0 0
• The risk minimizer for the Hamming loss is the \textit{marginal mode}:

\[ h^*_i(x) = \arg \max_{y_i \in \{0,1\}} P(y_i | x), \quad i = 1, \ldots, m, \]

while for the subset 0/1 loss is
The risk minimizer for the Hamming loss is the **marginal mode**:

\[ h_i^*(x) = \arg \max_{y_i \in \{0,1\}} P(y_i | x), \quad i = 1, \ldots, m, \]

while for the subset 0/1 loss is the **joint mode**:

\[ h^*(x) = \arg \max_{y \in \mathcal{Y}} P(y | x). \]
Hamming loss vs. subset 0/1 loss

- The risk minimizer for the Hamming loss is the **marginal mode**:

  
  \[ h_i^*(x) = \arg \max_{y_i \in \{0,1\}} P(y_i \mid x), \quad i = 1, \ldots, m, \]

  while for the subset 0/1 loss is the **joint mode**:

  \[ h^*(x) = \arg \max_{y \in \mathcal{Y}} P(y \mid x). \]

- Marginal mode vs. joint mode.

<table>
<thead>
<tr>
<th>y</th>
<th>P(y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0</td>
<td>0.30</td>
</tr>
<tr>
<td>0 1 1 1</td>
<td>0.17</td>
</tr>
<tr>
<td>1 0 1 1</td>
<td>0.18</td>
</tr>
<tr>
<td>1 1 0 1</td>
<td>0.17</td>
</tr>
<tr>
<td>1 1 1 0</td>
<td>0.18</td>
</tr>
</tbody>
</table>

  Marginal mode: 1 1 1 1

  Joint mode: 0 0 0 0
The risk minimizers of Hamming and subset 0/1 loss are different: marginal mode vs. joint mode.
• The risk minimizers of Hamming and subset 0/1 loss are different: marginal mode vs. joint mode.

• Under specific conditions, like label independence or high probability of the joint mode ($> 0.5$), these two risk minimizers are equivalent.
Relations between losses

- The risk minimizers of Hamming and subset 0/1 loss are different: marginal mode vs. joint mode.
- Under specific conditions, like label independence or high probability of the joint mode ($> 0.5$), these two risk minimizers are equivalent.
- The risks of these loss functions are mutually upper bounded.
Relations between losses

- The risk minimizers of Hamming and subset 0/1 loss are different: marginal mode vs. joint mode.
- Under specific conditions, like label independence or high probability of the joint mode (> 0.5), these two risk minimizers are equivalent.
- The risks of these loss functions are mutually upper bounded.
- Minimization of the subset 0/1 loss may cause a high regret for the Hamming loss and vice versa.
BR vs. LP

• Binary relevance (BR)
  ▶ BR is consistent for Hamming loss without any additional assumptions on label (in)dependence.
  ▶ If this would not be true, then we could not optimally solve binary classification problems!!!
  ▶ For other losses, one should take additional assumptions:
    • For subset 0/1 loss: label independence, high probability of the joint mode (> 0.5), . . .
  ▶ Learning and inference is linear in $m$ (however, faster algorithms exist).
• Binary relevance (BR)

- BR is consistent for Hamming loss without any additional assumptions on label (in)dependence.
- If this would not be true, then we could not optimally solve binary classification problems!!!
- For other losses, one should take additional assumptions:
  - For subset 0/1 loss: label independence, high probability of the joint mode (>0.5), . . .

- Learning and inference is linear in $m$ (however, faster algorithms exist).
• Binary relevance (BR)
  ▶ BR is **consistent** for Hamming loss **without** any additional assumptions on **label (in)dependence**.
• Binary relevance (BR)
  ▶ BR is **consistent** for Hamming loss **without** any additional assumptions on **label (in)dependence**.
  ▶ If this would not be true, then **we could not optimally solve binary classification problems!!!
BR vs. LP

• Binary relevance (BR)
  ▶ BR is consistent for Hamming loss without any additional assumptions on label (in)dependence.
  ▶ If this would not be true, then we could not optimally solve binary classification problems!!!
  ▶ For other losses, one should take additional assumptions:
    • For subset 0/1 loss: label independence, high probability of the joint mode (> 0.5), …
• Binary relevance (BR)
  ▶ BR is **consistent** for Hamming loss **without** any additional assumptions on label (in)dependence.
  ▶ If this would not be true, then we could not optimally solve binary classification problems!!!
  ▶ For other losses, one should take **additional assumptions**:  
    • For subset 0/1 loss: label independence, high probability of the joint mode (> 0.5), …
  ▶ Learning and inference is **linear** in $m$ (however, faster algorithms exist).
• Label powerset (LP)

LP is consistent for the subset 0/1 loss. In its basic formulation it is not consistent for Hamming loss. However, if used with a probabilistic multi-class classifier, it estimates the joint conditional distribution for a given $x$: inference for any loss would be then possible. Similarly, by reducing to cost-sensitive multi-class classification LP can be used with almost any loss function. LP may gain from the implicit expansion of the feature or hypothesis space. Unfortunately, learning and inference is basically exponential in $m$ (however, this complexity is constrained by the number of training examples).
• Label powerset (LP)
  ▶ LP is consistent for the subset 0/1 loss.
BR vs. LP

- Label powerset (LP)
  - LP is **consistent** for the subset 0/1 loss.
  - In its basic formulation it is **not consistent** for Hamming loss.
BR vs. LP

- Label powerset (LP)
  - LP is *consistent* for the subset 0/1 loss.
  - In its basic formulation it is *not consistent* for Hamming loss.
  - However, if used with a probabilistic multi-class classifier, it estimates the joint conditional distribution for a given $x$: inference for *any loss* would be then possible.
BR vs. LP

• Label powerset (LP)
  ▶ LP is **consistent** for the subset 0/1 loss.
  ▶ In its basic formulation it is **not consistent** for Hamming loss.
  ▶ However, if used with a probabilistic multi-class classifier, it estimates the joint conditional distribution for a given $x$: inference for **any loss** would be then possible.
  ▶ Similarly, by reducing to cost-sensitive multi-class classification LP can be used with **almost any loss function**.
BR vs. LP

- Label powerset (LP)
  - LP is **consistent** for the subset 0/1 loss.
  - In its basic formulation it is **not consistent** for Hamming loss.
  - However, if used with a probabilistic multi-class classifier, it estimates the joint conditional distribution for a given \( x \): inference for **any loss** would be then possible.
  - Similarly, by reducing to cost-sensitive multi-class classification LP can be used with **almost any loss function**.
  - LP may gain from the implicit expansion of the **feature** or **hypothesis space**.
BR vs. LP

- Label powerset (LP)
  - LP is **consistent** for the subset 0/1 loss.
  - In its basic formulation it is **not consistent** for Hamming loss.
  - However, if used with a probabilistic multi-class classifier, it estimates the joint conditional distribution for a given $x$: inference for **any loss** would be then possible.
  - Similarly, by reducing to cost-sensitive multi-class classification LP can be used with **almost any loss function**.
  - LP may gain from the implicit expansion of the **feature** or **hypothesis space**.
  - Unfortunately, learning and inference is basically **exponential** in $m$ (however, this complexity is constrained by the number of training examples).
Outline

1. Multi-label classification
2. Simple approaches to multi-label classification
3. Beyond simple approaches
4. Maximization of the F-measure
5. Rank loss minimization
6. Summary
Beyond LP

- **Classical multi-class classification algorithms:**
  - $k$-nearest neighbors,
  - Decision trees,
  - Logistic regression,
  - Multi-class SVMs,
  - ...

- **Reduction algorithms:**
  - 1 vs All,
  - 1 vs 1 and Weighted All-Pairs (WAP),
  - Directed acyclic graphs (DAG),
  - ECOC, PECOC, SECOC,
  - Filter Trees,
  - ...

- **Can we adapt these algorithms to multi-label classification and different task losses in a more direct way?**
Beyond LP

- Naive reduction to 1 vs. All:

\[(x, y) \rightarrow (x, y = \text{metaclass}(y))\]
Beyond LP

- Naive reduction to 1 vs. All:

\[(x, y) \rightarrow (x, y = \text{metaclass}(y))\]

- Reduction of **multi-class classification** to **binary classification**:

\[(x, y = \text{metaclass}(y)) \rightarrow \{(x, y, 1)\} \cup \{(x, y', 0) : \forall y' \neq y\}\]
• Naive reduction to 1 vs. All:

\[(x, y) \mapsto (x, y = \text{metaclass}(y))\]

• Reduction of **multi-class classification** to **binary classification**:

\[(x, y = \text{metaclass}(y)) \mapsto \{(x, y, 1)\} \cup \{(x, y', 0) : \forall y' \neq y\}\]

• But we can reduce directly **multi-label classification** to **binary classification**:

\[(x, y) \mapsto \{(x, y, 1)\} \cup \{(x, y', 0) : \forall y' \neq y\}\]
• Naive reduction to 1 vs. All:

\[(x, y) \rightarrow (x, y = \text{metaclass}(y))\]

• Reduction of multi-class classification to binary classification:

\[(x, y = \text{metaclass}(y)) \rightarrow \{(x, y, 1)\} \cup \{(x, y', 0) : \forall y' \neq y\}\]

• But we can reduce directly multi-label classification to binary classification:

\[(x, y) \rightarrow \{(x, y, 1)\} \cup \{(x, y', 0) : \forall y' \neq y\}\]

• We can exploit now the internal structure of label vectors!!!
The model can be given by a **scoring function** \( f(x, y) \).
Internal structure of classes

- The model can be given by a **scoring function** $f(x, y)$.
- Different forms of $f(x, y)$ are possible, for example:

$$f(x, y) = \sum_{i=1}^{m} f_i(x, y_i) + \sum_{y_k, y_l} f_{k,l}(y_k, y_l),$$

where the second term models pairwise interactions.

Prediction is given by:

$$h(x) = \arg \max_{y \in Y} f(x, y).$$
Internal structure of classes

- The model can be given by a **scoring function** $f(x, y)$.
- Different forms of $f(x, y)$ are possible, for example:

$$f(x, y) = \sum_{i=1}^{m} f_i(x, y_i) + \sum_{y_k, y_l} f_{k,l}(y_k, y_l),$$

where the second term models pairwise interactions.
Internal structure of classes

- The model can be given by a **scoring function** \( f(x, y) \).
- Different forms of \( f(x, y) \) are possible, for example:

\[
f(x, y) = \sum_{i=1}^{m} f_i(x, y_i) + \sum_{y_k, y_l} f_{k,l}(y_k, y_l),
\]

where the second term models pairwise interactions.
- Prediction is given by:

\[
h(x) = \arg \max_{y \in Y} f(x, y)
\]
Internal structure of classes

- Generalization of logistic regression and SVMs for $f(x, y)$:
  - Conditional random fields (CRFs),\(^3\)
  - Structured support vector machines (SSVMs).\(^4\)

---


CRFs and SSVMs

- CRFs use logistic loss as a surrogate:

\[
\tilde{\ell}_{\log}(y, x, f) = -\log P(y|x) = \log \left( \sum_{y \in Y} \exp(f(x, y)) \right) - f(x, y).
\]

- SSVMs minimize the structured hinge loss:

\[
\tilde{\ell}_{h}(y, x, f) = \max_{y' \in Y} \left\{ J_{y' \neq y} + f(x, y') \right\} - f(x, y).
\]
CRFs and SSVMs

- CRFs use **logistic loss** as a surrogate:

\[
\tilde{\ell}_{\log}(y, x, f) = -\log P(y|x) = \log \left( \sum_{y \in Y} \exp(f(x, y)) \right) - f(x, y).
\]

- SSVMs minimize the **structured hinge loss**:

\[
\tilde{\ell}_h(y, x, f) = \max_{y' \in Y} \{[y' \neq y] + f(x, y')\} - f(x, y).
\]
CRFs and SSVMs

- CRFs use **logistic loss** as a surrogate:

\[
\tilde{\ell}_{\text{log}}(y, x, f) = - \log P(y|x) = \log \left( \sum_{y \in Y} \exp(f(x, y)) \right) - f(x, y).
\]

- SSVMs minimize the **structured hinge loss**:

\[
\tilde{\ell}_h(y, x, f) = \max_{y' \in Y} \{ [y' \neq y] + f(x, y') \} - f(x, y).
\]

- SSVMs and CRFs are quite similar to each other:
  - max vs. soft-max
  - margin vs. no-margin
CRFs and SSVMs

- Follow the general LP strategy, but can exploit the internal structure of classes within scoring function $f(x, y)$.
- Convex optimization problem, but its hardness depends on the structure of $f(x, y)$.
- Similarly, the inference (also known as decoding problem) is hard in the general case.
- For sequence and tree structures, the problem can be solved in polynomial time.
CRFs and SSVMs for different task losses

• In SSVMs, task loss $\ell(y, y')$ can be used for **margin rescaling**:

$$\tilde{\ell}_h(y, x, f) = \max_{y' \in \mathcal{Y}} \{\ell(y, y') + f(x, y')\} - f(x, y).$$

• SSVMs with Hamming loss and

$$f(x, y) = \sum_{i=1}^{m} f_i(x, y_i)$$

decompose to BR with SVMs.

• In general SSVMs are inconsistent.\(^5\)

---


CRFs and SSVMs for different task losses

• CRFs are tailored for the subset 0/1 loss and cannot directly take other task losses into account.

• CRFs with the scoring function of the form

\[
f(x, y) = \sum_{i=1}^{m} f_i(x, y_i)
\]

minimize Hamming loss (→ BR with logistic regression).

• Some works on incorporating margin into CRFs.\(^6\)


SSVMs vs. BR

Table: SSVMs with pairwise term\textsuperscript{7} vs. BR with LR\textsuperscript{8}.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>SSVM Best</th>
<th>BR LR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scene</td>
<td>0.101±.003</td>
<td>0.102±.003</td>
</tr>
<tr>
<td>Yeast</td>
<td>0.202±.005</td>
<td>0.199±.005</td>
</tr>
<tr>
<td>Synth1</td>
<td>0.069±.001</td>
<td>0.067±.002</td>
</tr>
<tr>
<td>Synth2</td>
<td>0.058±.001</td>
<td>0.084±.001</td>
</tr>
</tbody>
</table>

- There is almost no difference between both algorithms.

\textsuperscript{7} Thomas Finley and Thorsten Joachims. Training structural SVMs when exact inference is intractable. In ICML. Omnipress, 2008

Probabilistic classifier chains

- Probabilistic classifier chains (PCCs)\(^9\) are an efficient reduction method similar to conditional probability trees.\(^{10}\)
- They estimate the joint conditional distribution \(P(y \mid x)\) as CRFs.
- The underlying idea is to repeatedly apply the **product rule of probability**:

\[
P(y \mid x) = \prod_{i=1}^{m} P(y_i \mid x, y_1, \ldots, y_{i-1}).
\]

---


Probabilistic classifier chains

• They follow a reduction to a sequence of subproblems:

\[(x, y) \rightarrow (x', y_1, \ldots, y_{i-1}, y = y_i), \quad i = 1, \ldots, m\]

• Each subproblem is solved independently by a probabilistic classifier estimating

\[P(y_i = 1 | x').\]

• By using linear models in each task independently, the overall scoring function takes the form:

\[f(x, y) = \sum_{i=1}^{m} f_i(x, y_i) + \sum_{y_k, y_l} f_{k,l}(y_k, y_l)\]
• Inference relies on exploiting a probability tree being the result of PCC:

\[
\begin{align*}
P(y_1 = 0 | x) &= 0.4 \\
P(y_1 = 1 | x) &= 0.6
\end{align*}
\]

\[
\begin{align*}
P(y_2 = 0 | y_1 = 0, x) &= 0.0 & P(y_2 = 1 | y_1 = 0, x) &= 1.0 & P(y_2 = 0 | y_1 = 1, x) &= 0.4 & P(y_2 = 1 | y_1 = 1, x) &= 0.6 \\
P(y = (0, 0) | x) &= 0 & P(y = (0, 1) | x) &= 0.4 & P(y = (1, 0) | x) &= 0.24 & P(y = (1, 1) | x) &= 0.36
\end{align*}
\]

• For subset 0/1 loss one needs to find \( h(x) = \arg \max_{y \in \mathcal{Y}} P(y | x). \)

• Greedy and approximate search techniques with guarantees exist.\(^{11}\)


Probabilistic classifier chains

- Inference relies on exploiting a probability tree being the result of PCC:

```
\begin{array}{cccc}
\text{y}_1 &=& 0 & \text{y}_1 &=& 1 \\
\text{y}_2 &=& 0 & \text{y}_2 &=& 1 \\
P(\text{y}_1=0 \g x) &=& 0.4 & P(\text{y}_1=1 \g x) &=& 0.6 \\
\end{array}
```

- Other losses: compute the prediction on a sample from $P(\text{y} \g x)$.\footnote{K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier. An analysis of chaining in multi-label classification. In ECAI, 2012}

- Sampling can be easily performed by using the probability tree.
Table: PCC vs. SSVMs on Hamming loss and PCC vs. BR on subset 0/1 loss.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>PCC</th>
<th>SSVM</th>
<th>Best</th>
<th>PCC</th>
<th>BR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hamming loss</td>
<td></td>
<td></td>
<td>subset 0/1 loss</td>
<td></td>
</tr>
<tr>
<td>Scene</td>
<td>0.104±.004</td>
<td>0.101±.003</td>
<td>0.385±.014</td>
<td>0.509±.014</td>
<td></td>
</tr>
<tr>
<td>Yeast</td>
<td>0.203±.005</td>
<td>0.202±.005</td>
<td>0.761±.014</td>
<td>0.842±.012</td>
<td></td>
</tr>
<tr>
<td>Synth1</td>
<td>0.067±.001</td>
<td>0.069±.001</td>
<td>0.239±.006</td>
<td>0.240±.006</td>
<td></td>
</tr>
<tr>
<td>Synth2</td>
<td>0.000±.000</td>
<td>0.058±.001</td>
<td>0.000±.000</td>
<td>0.832±.004</td>
<td></td>
</tr>
<tr>
<td>Reuters</td>
<td>0.060±.002</td>
<td>0.045±.001</td>
<td>0.598±.009</td>
<td>0.689±.008</td>
<td></td>
</tr>
<tr>
<td>Mediamill</td>
<td>0.172±.001</td>
<td>0.182±.001</td>
<td>0.885±.003</td>
<td>0.902±.003</td>
<td></td>
</tr>
</tbody>
</table>
Outline

1. Multi-label classification
2. Simple approaches to multi-label classification
3. Beyond simple approaches
4. Maximization of the F-measure
5. Rank loss minimization
6. Summary
Maximization of the F-measure

- Applications: Information retrieval, document tagging, and NLP.

- JRS 2012 Data Mining Competition: Indexing documents from MEDLINE or PubMed Central databases with concepts from the Medical Subject Headings ontology.
Maximization of the F-measure

- The $F_\beta$-measure-based loss function ($F_\beta$-loss):

$$\ell_{F_\beta}(y, h(x)) = 1 - F_\beta(y, h(x))$$

$$= 1 - \frac{(1 + \beta^2) \sum_{i=1}^{m} y_i h_i(x)}{\beta^2 \sum_{i=1}^{m} y_i + \sum_{i=1}^{m} h_i(x)} \in [0, 1].$$

- Provides a **better balance** between relevant and irrelevant labels.
- However, it **is not easy** to optimize.
SSVMs for $F_\beta$-based loss

- SSVMs can be used to minimize $F_\beta$-based loss.
- Rescale the margin by $\ell_F(y, y')$.
- Two algorithms:\(^{12}\)
  
  **RML**
  
  No label interactions:
  
  \[
  f(y, x) = \sum_{i=1}^{m} f_i(y_i, x)
  \]
  
  Quadratic learning and linear prediction

  **SML**
  
  Submodular interactions:
  
  \[
  f(y, x) = \sum_{i=1}^{m} f_i(y_i, x) + \sum_{y_k, y_l} f_{k,l}(y_k, y_l)
  \]
  
  More complex (graph-cut and approximate algorithms)

- Both are inconsistent.


Plug-in rule approach

- Plug estimates of required parameters into the Bayes classifier:

\[
\begin{align*}
    h^* &= \arg \min_{h \in \mathcal{Y}} \mathbb{E} \left[ \ell_{F_\beta}(Y, h) \right] \\
    &= \arg \max_{h \in \mathcal{Y}} \sum_{y \in \mathcal{Y}} P(y) \frac{(\beta + 1) \sum_{i=1}^{m} y_i h_i}{\beta^2 \sum_{i=1}^{m} y_i + \sum_{i=1}^{m} h_i}
\end{align*}
\]

- No closed form solution for this optimization problem.
- The problem cannot be solved naively by brute-force search:
  - This would require to check all possible combinations of labels \(2^m\)
  - To sum over \(2^m\) number of elements for computing the expected value.
  - The number of parameters to be estimated \(P(y)\) is \(2^m\).

---

Plug-in rule approach

• Approximation needed?

Plug-in rule approach

- **Approximation needed?** Not really. The exact solution is tractable!

---


Plug-in rule approach

- **Approximation needed?** Not really. The exact solution is tractable!

**LFP:**
- Assumes label independence.
- Linear number of parameters: $P(y_i = 1)$.
- Inference based on dynamic programming.\(^{14}\)
- Reduction to LR for each label.

**EFP:**
- No assumptions.
- Quadratic number of parameters: $P(y_i = 1, s = \sum_i y_i)$.
- Inference based on matrix multiplication and top $k$ selection.\(^{15}\)
- Reduction to multinomial LR for each label.

- **EFP is consistent.**\(^ {16}\)

---


Maximization of the F-measure

<table>
<thead>
<tr>
<th>Dataset</th>
<th>F1-measure [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IMAGE</strong></td>
<td></td>
</tr>
<tr>
<td>EFP</td>
<td>59.77</td>
</tr>
<tr>
<td>LFP</td>
<td>58.86</td>
</tr>
<tr>
<td>RML</td>
<td>57.49</td>
</tr>
<tr>
<td>SML</td>
<td>56.99</td>
</tr>
<tr>
<td>BR</td>
<td>43.63</td>
</tr>
<tr>
<td><strong>SCENE</strong></td>
<td></td>
</tr>
<tr>
<td>EFP</td>
<td>74.44</td>
</tr>
<tr>
<td>LFP</td>
<td>74.38</td>
</tr>
<tr>
<td>RML</td>
<td>73.92</td>
</tr>
<tr>
<td>SML</td>
<td>68.5</td>
</tr>
<tr>
<td>BR</td>
<td>55.73</td>
</tr>
<tr>
<td><strong>YEAST</strong></td>
<td></td>
</tr>
<tr>
<td>EFP</td>
<td>65.47</td>
</tr>
<tr>
<td>LFP</td>
<td>65.02</td>
</tr>
<tr>
<td>RML</td>
<td>64.78</td>
</tr>
<tr>
<td>SML</td>
<td>63.96</td>
</tr>
<tr>
<td>BR</td>
<td>60.59</td>
</tr>
<tr>
<td><strong>MEDICAL</strong></td>
<td></td>
</tr>
<tr>
<td>EFP</td>
<td>80.39</td>
</tr>
<tr>
<td>LFP</td>
<td>81.27</td>
</tr>
<tr>
<td>RML</td>
<td>80.63</td>
</tr>
<tr>
<td>SML</td>
<td>67.9</td>
</tr>
<tr>
<td>BR</td>
<td>70.19</td>
</tr>
<tr>
<td><strong>ENRON</strong></td>
<td></td>
</tr>
<tr>
<td>EFP</td>
<td>61.04</td>
</tr>
<tr>
<td>LFP</td>
<td>56.86</td>
</tr>
<tr>
<td>RML</td>
<td>57.69</td>
</tr>
<tr>
<td>SML</td>
<td>54.61</td>
</tr>
<tr>
<td>BR</td>
<td>55.49</td>
</tr>
<tr>
<td><strong>MEDIAMILL</strong></td>
<td></td>
</tr>
<tr>
<td>EFP</td>
<td>55.16</td>
</tr>
<tr>
<td>LFP</td>
<td>55.15</td>
</tr>
<tr>
<td>RML</td>
<td>49.35</td>
</tr>
<tr>
<td>SML</td>
<td>50.02</td>
</tr>
<tr>
<td>BR</td>
<td>51.21</td>
</tr>
</tbody>
</table>
Outline

1. Multi-label classification
2. Simple approaches to multi-label classification
3. Beyond simple approaches
4. Maximization of the F-measure
5. Rank loss minimization
6. Summary
Multi-label ranking

Multi-label classification

- politics: 0
- economy: 0
- business: 0
- sport: 1
- tennis: 1
- soccer: 0
- show-business: 0
- celebrities: 1
- England: 1
- USA: 1
- Poland: 1
- Lithuania: 0
Multi-label ranking

Serena romps to fifth Wimbledon title against brave Radwanska

By Paul Gittings, CNN
July 7, 2012 -- Updated 22:20 GMT (06:20 HKT)

Women’s singles Wimbledon Championship

STORY HIGHLIGHTS
• Serena Williams wins fifth Wimbledon crown
• American beats Agnieszka Radwanska of Poland 6-1 5-7 6-2
• Radwanska battles respiratory

(CNN) -- Serena Williams fended off a stirring fightback from Agnieszka Radwanska to win her fifth Wimbledon singles title with a 6-1 5-7 6-2 victory Saturday.

It was the 30-year-old American’s 14th grand slam crown and her first since winning at the All England Club in 2010, but Poland’s Radwanska made her fight every inch of the way.

Multi-label ranking

tennis
sport
England
Poland
USA
politics
Multi-label ranking

- **Ranking loss:**

\[ \ell_{\text{rnk}}(y, f) = w(y) \sum_{(i,j): y_i > y_j} \left( \mathbb{[} f_i(x) < f_j(x) \mathbb{]} + \frac{1}{2} \mathbb{[} f_i(x) = f_j(x) \mathbb{]} \right), \]

where \( w(y) < w_{\text{max}} \) is a weight function.

<table>
<thead>
<tr>
<th>( X_1 )</th>
<th>( X_2 )</th>
<th>( Y_1 )</th>
<th>( Y_2 )</th>
<th>...</th>
<th>( Y_m )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x )</td>
<td>4.0</td>
<td>2.5</td>
<td>1</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>( h_2 )</td>
<td>( &gt; )</td>
<td>( h_1 )</td>
<td>( &gt; )</td>
<td>...</td>
<td>( &gt; )</td>
</tr>
</tbody>
</table>
Multi-label ranking

- **Ranking loss:**

\[
\ell_{\text{rnk}}(y, f) = w(y) \sum_{(i,j): y_i > y_j} \left( [f_i(x) < f_j(x)] + \frac{1}{2} [f_i(x) = f_j(x)] \right),
\]

where \( w(y) < w_{\text{max}} \) is a weight function.

The weight function \( w(y) \) is usually used to normalize the range of rank loss to \([0, 1]\):

\[
w(y) = \frac{1}{n_+ n_-},
\]

i.e., it is equal to the inverse of the total number of pairwise comparisons between labels.
Pairwise surrogate losses

- The most intuitive approach is to use pairwise convex surrogate losses of the form

\[
\tilde{\ell}_\phi(y, f) = \sum_{(i,j): y_i > y_j} w(y) \phi(f_i - f_j),
\]

where \( \phi \) is

- an exponential function (BoosTexter)\(^{17} \): \( \phi(f) = e^{-f} \),
- logistic function (LLLR)\(^{18} \): \( \phi(f) = \log(1 + e^{-f}) \),
- or hinge function (RankSVM)\(^{19} \): \( \phi(f) = \max(0, 1 - f) \).

---


Multi-label ranking

- This approach is, however, **inconsistent** for the most commonly used convex surrogates.\(^{20}\)
- The **consistent** classifier can be, however, obtained by using univariate loss functions\(^{21}\) ...
• The Bayes ranker can be obtained by sorting labels according to:

\[ \Delta^1_i = \sum_{y: y_i = 1} w(y) P(y \mid x). \]

• For \( w(y) \equiv 1 \), \( \Delta^u_i \) reduces to **marginal probabilities** \( P(y_i = u \mid x) \).

• The solution can be obtained with BR or its weighted variant in a general case.
Reduction to weighted binary relevance

- Consider the sum of univariate (weighted) losses:

\[ \tilde{\ell}_{\text{exp}}(y, f) = w(y) \sum_{i=1}^{m} e^{-y'f_i}, \]

\[ \tilde{\ell}_{\text{log}}(y, f) = w(y) \sum_{i=1}^{m} \log \left(1 + e^{-y'f_i}\right). \]

where \( y' = 2y_i - 1 \).

- The risk minimizer of these losses is:

\[ f_i^*(x) = \frac{1}{c} \log \frac{\Delta_1^1}{\Delta_0^i} = \frac{1}{c} \log \frac{\Delta_1^1}{W - \Delta_1^i}, \]

which is a strictly increasing transformation of \( \Delta_1^i \), where

\[ W = \mathbb{E}_y[w(y) \mid x] = \sum_y w(y)P(y \mid x). \]
Reduction to weighted binary relevance

- **Vertical reduction**: Solving $m$ independent classification problems.
- Standard algorithms, like AdaBoost and logistic regression, can be adapted to this setting.
- AdaBoost.MH follows this approach for $w = 1$.\(^\text{22}\)
- Besides its **simplicity** and **efficiency**, this approach is **consistent** (regret bounds have also been derived).\(^\text{23}\)


Weighted binary relevance

Figure: WBR–LR vs. LLLR. Left: independent data. Right: dependent data.

- **Label independence**: the methods perform more or less en par.
- **Label dependence**: WBR shows small but consistent improvements.
### Benchmark data

Table: WBR-AdaBoost vs. AdaBoost.MR (left) and WBR-LR vs LLLR (right).

<table>
<thead>
<tr>
<th>DATASET</th>
<th>AB.MR</th>
<th>WBR-AB</th>
<th>LLLR</th>
<th>WBR-LR</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMAGE</td>
<td>0.2081</td>
<td>0.2041</td>
<td>0.2047</td>
<td>0.2065</td>
</tr>
<tr>
<td>EMOTIONS</td>
<td>0.1703</td>
<td>0.1699</td>
<td>0.1743</td>
<td>0.1657</td>
</tr>
<tr>
<td>SCENE</td>
<td>0.0720</td>
<td>0.0792</td>
<td>0.0861</td>
<td>0.0793</td>
</tr>
<tr>
<td>YEAST</td>
<td>0.2072</td>
<td>0.1820</td>
<td>0.1728</td>
<td>0.1736</td>
</tr>
<tr>
<td>MEDIAMILL</td>
<td>0.0665</td>
<td>0.0609</td>
<td>0.0614</td>
<td>0.0472</td>
</tr>
</tbody>
</table>

- WBR is at least competitive to state-of-the-art algorithms defined on pairwise surrogates.
Outline

1 Multi-label classification
2 Simple approaches to multi-label classification
3 Beyond simple approaches
4 Maximization of the F-measure
5 Rank loss minimization
6 Summary
Open challenges

- Learning and inference algorithms for any task loss and output structure.
- Consistency of the algorithms.
- Large-scale datasets: number of instances, features, and labels.
Conclusions

• Take-away message:
  ▶ Two main issues: loss minimization and label dependence.
  ▶ Two main approaches: surrogate loss minimization and reduction.
  ▶ Consistency of algorithms.
  ▶ High regret between solutions for different losses.
  ▶ Proper modeling of label dependence for different loss functions.
  ▶ Be careful with empirical evaluations.
  ▶ Independent models can perform quite well.

• For more check:

  http://www.cs.put.poznan.pl/kdembczynski