Label Space Coding for Multi-label Classification

Hsuan-Tien Lin
National Taiwan University
Talk at IIS Sinica, April 26, 2012

joint works with

Farbound Tai (MLD Workshop 2011, NC Journal 2012) &
Chun-Sung Ferng (ACML Conference 2011, NTU Thesis 2012)
Multi-label Classification

Which Fruit?

? apple orange strawberry kiwi

multi-class classification:
classify input (picture) to **one category** (label)
Which Fruits?

?: \{orange, strawberry, kiwi\}

apple
orange
strawberry
kiwi

multi-label classification: classify input to multiple (or no) categories
Powerset: Multi-label Classification via Multi-class

Multi-class w/ $L = 4$ classes

- 4 possible outcomes
  \{a, o, s, k\}

Multi-label w/ $L = 4$ classes

- $2^4 = 16$ possible outcomes
  \[2^{\{a, o, s, k\}}\]

\[\uparrow\]

\[\{\phi, a, o, s, k, ao, as, ak, os, ok, sk, aos, aok, ask, osk, aosk\}\]

- **Powerset** approach: reduction to multi-class classification
- difficulties for large $L$:
  - **computation** (super-large $2^L$)
    — hard to construct classifier
  - **sparsity** (no example for some of $2^L$)
    — hard to discover hidden combination

**Powerset**: feasible only for small $L$ with enough examples for every combination
Multi-label Classification

What Tags?

?: \{machine learning, data structure, data mining, object oriented programming, artificial intelligence, compiler, architecture, chemistry, textbook, children book, ... etc. \}

another multi-label classification problem: tagging input to multiple categories
Binary Relevance: Multi-label Classification via Yes/No

**Binary Classification**

\{yes, no\}

**Multi-label w/ \( L \) classes: \( L \) yes/no questions**

- machine learning (Y), data structure (N), data mining (Y), OOP (N), AI (Y), compiler (N), architecture (N), chemistry (N), textbook (Y), children book (N), etc.

- **Binary Relevance** approach:
  - reduction to multiple isolated binary classification

- disadvantages:
  - **isolation**—hidden relations not exploited (e.g. ML and DM highly correlated, ML subset of AI, textbook & children book disjoint)
  - **unbalancedness**—few yes, many no

**Binary Relevance**: simple (& good) benchmark with known disadvantages
Multi-label Classification

Multi-label Classification Setup

Given

\(N\) examples (input \(x_n\), label-set \(y_n\)) \(\in \mathcal{X} \times 2^{\{1,2,\ldots,L\}}\)

- fruits: \(\mathcal{X} = \text{encoding(pictures)}\), \(y_n \subseteq \{1,2,\ldots,4\}\)
- tags: \(\mathcal{X} = \text{encoding(merchandise)}\), \(y_n \subseteq \{1,2,\ldots,L\}\)

Goal

a multi-label classifier \(g(x)\) that closely predicts the label-set \(\mathcal{Y}\) associated with some unseen inputs \(x\) (by exploiting hidden relations/combinations between labels)

- 0/1 loss: any discrepancy \([g(x) \neq \mathcal{Y}]\)
- Hamming loss: averaged symmetric difference \(\frac{1}{L} |g(x) \triangle \mathcal{Y}|\)

multi-label classification: hot and important
Topics in this Talk

1. **Coding/Geometric View of Multi-label Classification** — **unify** existing algorithms w/ intuitive explanations
2. **Compression Coding** — **condense** for efficiency — capture hidden correlation
3. **Error-correction Coding** — **expand** for accuracy — capture hidden combination
Coding/Geometric View of Multi-label Classification
## Coding/Geometric View of Multi-label Classification

### From Label-set to Coding View

<table>
<thead>
<tr>
<th>label set</th>
<th>apple</th>
<th>orange</th>
<th>strawberry</th>
<th>binary code</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{Y}_1 = {o}$</td>
<td>0 (N)</td>
<td>1 (Y)</td>
<td>0 (N)</td>
<td>$y_1 = [0, 1, 0]$</td>
</tr>
<tr>
<td>$\mathcal{Y}_2 = {a, o}$</td>
<td>1 (Y)</td>
<td>1 (Y)</td>
<td>0 (N)</td>
<td>$y_2 = [1, 1, 0]$</td>
</tr>
<tr>
<td>$\mathcal{Y}_3 = {a, s}$</td>
<td>1 (Y)</td>
<td>0 (N)</td>
<td>1 (Y)</td>
<td>$y_3 = [1, 0, 1]$</td>
</tr>
<tr>
<td>$\mathcal{Y}_4 = {o}$</td>
<td>0 (N)</td>
<td>1 (Y)</td>
<td>0 (N)</td>
<td>$y_4 = [0, 1, 0]$</td>
</tr>
<tr>
<td>$\mathcal{Y}_5 = {}$</td>
<td>0 (N)</td>
<td>0 (N)</td>
<td>0 (N)</td>
<td>$y_5 = [0, 0, 0]$</td>
</tr>
</tbody>
</table>

subset $\mathcal{Y}$ of $2^{\{1,2,\ldots,L\}} \leftrightarrow$ length-$L$ binary code $y$
Coding/Geometric View of Multi-label Classification

Existing Approach: Compressive Sensing

General Compressive Sensing

sparse (many 0) binary vectors \( y \in \{0, 1\}^L \) can be robustly compressed by projecting to \( M \ll L \) basis vectors \( \{p_1, p_2, \cdots, p_M\} \)

Compressive Sensing for Multi-label Classification (Hsu et al., 2009)

1. **compress**: transform \( \{(x_n, y_n)\} \) to \( \{(x_n, c_n)\} \) by \( c_n = Py_n \) with some \( M \) by \( L \) random matrix \( P = [p_1, p_2, \cdots, p_M]^T \)

2. **learn**: get regression function \( r(x) \) from \( x_n \) to \( c_n \)

3. **decode**: \( g(x) = \text{find closest sparse binary vector to } P^T r(x) \)

Compressive Sensing:

- reduction to multi-output regression w/ codewords \( c \)
  - efficient in training: *random projection* w/ \( M \ll L \)
  - inefficient in testing: time-consuming decoding
From Coding View to Geometric View

<table>
<thead>
<tr>
<th>Label Set</th>
<th>Binary Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{Y}_1 = {o}$</td>
<td>$y_1 = [0, 1, 0]$</td>
</tr>
<tr>
<td>$\mathcal{Y}_2 = {a, o}$</td>
<td>$y_2 = [1, 1, 0]$</td>
</tr>
<tr>
<td>$\mathcal{Y}_3 = {a, s}$</td>
<td>$y_3 = [1, 0, 1]$</td>
</tr>
<tr>
<td>$\mathcal{Y}_4 = {o}$</td>
<td>$y_4 = [0, 1, 0]$</td>
</tr>
<tr>
<td>$\mathcal{Y}_5 = {}$</td>
<td>$y_5 = [0, 0, 0]$</td>
</tr>
</tbody>
</table>

length-$L$ binary code $\Leftrightarrow$ vertex of hypercube $\{0, 1\}^L$
Geometric Interpretation of Powerset

Powerset: directly classify to the **vertices** of hypercube
Binary Relevance: project to the **natural axes** & classify
Compressive Sensing:

- project to **random flat** (linear subspace)
- learn “on” the flat; decode to **closest sparse vertex**

other (better) flat? other (faster) decoding?
Compression Coding (Using Geometry)

A Novel Approach for Label Space Compression

- algorithmic: scheme for **fast decoding**
- theoretical: justification for **best flat**
- practical: **significantly better performance** than compressive sensing (& binary relevance)

will now introduce the key ideas behind the approach
Compressive Sensing Revisited

1. **compress**: transform \( \{(x_n, y_n)\} \) to \( \{(x_n, c_n)\} \) by \( c_n = Py_n \) with some \( M \) by \( L \) random matrix \( P \)

2. **learn**: get regression function \( r(x) \) from \( x_n \) to \( c_n \)

3. **decode**: \( g(x) = \) find closest sparse binary vector to \( P^Tr(x) \)

- find closest **sparse** binary vector to \( \tilde{y} \): slow optimization of \( \ell_1 \)-regularized objective
- find closest **any** binary vector to \( \tilde{y} \): fast

\[ g(x) = \text{round}(y) \]

**round-based decoding**: simple & faster alternative
Compressive Sensing Revisited

1. **compress**: transform \( \{(x_n, y_n)\} \) to \( \{(x_n, c_n)\} \) by \( c_n = Py_n \) with some \( M \) by \( L \) random matrix \( P \)

2. **learn**: get regression function \( r(x) \) from \( x_n \) to \( c_n \)

3. **decode**: \( g(x) = \) find closest sparse binary vector to \( P^T r(x) \)

- **random flat**: arbitrary directions
- **best flat**: principal directions

**principal directions/flat**: best approximation to vertices \( y_n \) during compression (why?)
Novel Theoretical Guarantee

Linear Transform + Regress + Round-based Decoding

Theorem (Tai and Lin, 2012)

If $g(x) = \text{round}(P^T r(x))$,

$$\frac{1}{L} |g(x) \triangle Y| \leq \text{const} \cdot \left( \|r(x) - c\|^2 \right)$$

- $\|r(x) - c\|^2$: prediction error from input to codeword
- $\|y - P^T c\|^2$: encoding error from vertex to codeword

principal directions/flat: best approximation to vertices $y_n$ during compression (indeed)
Proposed Approach: Principal Label Space Transform

From Compressive Sensing to **PLST**

1. **compress**: transform \( \{(x_n, y_n)\} \) to \( \{(x_n, c_n)\} \) by \( c_n = P(y_n - o) \) with the \( M \) by \( L \) principal matrix \( P \) and some reference point \( o \).

2. **learn**: get regression function \( r(x) \) from \( x_n \) to \( c_n \).

3. **decode**: \( g(x) = \text{round}(P^T r(x) + o) \).

- **reference point \( o \)**: allow flat not passing the origin
- **best \( o \) and \( P \)**:

\[
\begin{align*}
\min_{o, p_1, p_2, \ldots, p_M} \quad & \sum_{n=1}^{N} \left\| y_n - o - P^T P(y_n - o) \right\|^2 \\
\text{subject to} & \quad \text{orthonormal vectors } p_m
\end{align*}
\]
Solving for Principal Directions

\[
\begin{align*}
\min_{\mathbf{o}, \mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_M} & \quad \sum_{n=1}^{N} \left\| \mathbf{y}_n - \mathbf{o} - \mathbf{P}^T \mathbf{P} (\mathbf{y}_n - \mathbf{o}) \right\|^2 \\
\text{subject to} & \quad \text{orthonormal vectors } \mathbf{p}_m
\end{align*}
\]

- solution: **Principal Component Analysis** on \(\{\mathbf{y}_n\}_{n=1}^{N}\)
- best \(\mathbf{o}\): \(\frac{1}{N} \sum_{n=1}^{N} \mathbf{y}_n\)
- best \(\mathbf{p}_m\): top eigenvectors of \(\sum_{n=1}^{N} (\mathbf{y}_n - \mathbf{o})(\mathbf{y}_n - \mathbf{o})^T\)
- physical meaning behind \(\mathbf{p}_m\): key (linear) label correlations (e.g. like eigenface in face recognition)

**PLST:** reduction to multi-output regression by projecting to **key correlations**
Compression Coding (Using Geometry)

Hamming Loss Comparison: Full-BR, PLST & CS

- **PLST** better than **Full-BR**: fewer dimensions, similar (or better) performance
- **PLST** better than **CS**: faster, **better** performance
- Similar findings across **data sets** and **regression algorithms**
Semi-summary on PLST

- reduction to **multi-output regression**
- project to **principal directions** and capture key correlations
- efficient learning (**label space compression**)
- efficient decoding (**round-based**)
- sound theoretical guarantee + **good practical performance** (better than CS & BR)

**expansion** (channel coding) instead of compression ("lossy" source coding)? **YES!**

will start by reviewing an existing algorithm
Random $k$-labelsets

**Random $k$-labELsets** (Tsoumakas & Vlahavas, 2007)

- Training data: $L$ labels
- Pick $k$ labels
- Label powerset
- Predict $k$ labels
- Majority vote
- Prediction of $L$ labels

- $y = (1, 0, 0, 1)$
- \((y_1, y_3) = (1, 0)\)
- Label powerset
- \((	ilde{y}_1, 	ilde{y}_3) = (1, 0)\)

- More efficient than Powerset: many $2^k$ instead of $2^L$

- More robust than Powerset: hidden combinations detected

**RA$k$EL:**
- Reduction to many $2^k$-category classification tasks
Random $k$-labelsets from Coding View

**RAndom $k$-labELsets** (Tsoumakas & Vlahavas, 2007)

- encode ($y \rightarrow b$): repetition & permutation
- learn: $k$-label powerset, i.e. run Powerset on size-$k$ chunk of bits
- decode ($\tilde{b} \rightarrow \tilde{y}$): majority vote

\[ y = (1,0,0,1) \]
\[ b = (y_1, y_3, y_4, y_1, y_2, y_1) = (1,0,1,0,1) \]
\[ \text{2-label powerset} \]
\[ \tilde{b} = (y_1, y_3, y_4, y_1, y_2, y_1) = (1,0,1,0,0) \]
\[ \text{majority vote} \]
\[ \tilde{y} = (1,0,0,1) \]
Error-correction Coding

A Novel Framework for Label Space Error-correction

- algorithmic: generalize RAnEL and explain through coding view
- theoretical: link learning performance to error-correcting ability
- practical: explore choices of error-correcting code and obtain better performance than RAnEL (& binary relevance)

will now introduce the framework
Key Idea: Redundant Information

General Error-correcting Codes (ECC)

- Noisy channel:
  - Encode data redundantly
  - Commonly used in communication systems
  - Detect & correct errors after transmitting data over a noisy channel

ECC for Machine Learning (successful for multi-class classification)

- Learn redundant bits \( \Rightarrow \) correct prediction errors
Proposed Framework: Multi-labeling with ECC

- **encode** to add redundant information \( enc(\cdot): \{0, 1\}^L \rightarrow \{0, 1\}^M \)
- **decode** to locate most possible binary vector \( dec(\cdot): \{0, 1\}^M \rightarrow \{0, 1\}^L \)
- reduction to **larger multi-label classification** with labels \( b \)

**PLST:** \( M \ll L \) (works for large \( L \));
**MLECC:** \( M > L \) (works for small \( L \))
Theorem

Let $g(x) = \text{dec}(\tilde{b})$ with $\tilde{b} = h(x)$. Then,

\[
\left\lfloor g(x) \neq \mathcal{Y} \right\rfloor \leq \text{const.} \cdot \frac{\text{Hamming loss of } h(x)}{\text{ECC strength} + 1}.
\]

0/1 loss

PLST: principal directions + decent regression
MLECC: which ECC balances strength & difficulty?
Simplest ECC: Repetition Code

**Encoding:**  \( y \in \{0, 1\}^L \rightarrow b \in \{0, 1\}^M \)
- **repeat** each bit \( \frac{M}{L} \) times

\[
L = 4, \quad M = 28 : 1010 \rightarrow \underbrace{1111111}_{28/4 = 7} 000000011111110000000
\]
- permute the bits randomly

**Decoding:**  \( \tilde{b} \in \{0, 1\}^M \rightarrow \tilde{y} \in \{0, 1\}^L \)
- **majority vote** on each original bit

\( L = 4, \quad M = 28: \text{strength of repetition code (REP) = 3} \)

**RAkEL = REP + k-label powerset**
Slightly More Sophisticated: Hamming Code

**HAM(7, 4) Code**

- \( \{0, 1\}^4 \rightarrow \{0, 1\}^7 \) via adding 3 **parity bits**
  - physical meaning: **label combinations**
- \( b_4 = y_0 \oplus y_1 \oplus y_3, \ b_5 = y_0 \oplus y_2 \oplus y_3, \ b_6 = y_1 \oplus y_2 \oplus y_3 \)
- e.g. 1011 → 1011010
- strength = 1 (weak)

**Our Proposed Code: Hamming on Repetition (HAMR)**

\[
\{0, 1\}^L \xrightarrow{\text{REP}} \{0, 1\}^{4M/7} \xrightarrow{\text{HAM(7, 4) on each 4-bit block}} \{0, 1\}^{7M/7}
\]

\( L = 4, \ M = 28: \) strength of HAMR = 4 **better than REP!**

**HAMR + \( k\)-label powerset:**
improvement of RAKEL on **code strength**
Bose-Chaudhuri-Hocquenghem Code (BCH)
- modern code in **CD players**
- sophisticated extension of Hamming, with **more parity bits**
- codeword length $M = 2^p - 1$ for $p \in \mathbb{N}$
- $L = 4$, $M = 31$, strength of BCH = 5

Low-density Parity-check Code (LDPC)
- modern code for **satellite communication**
- connect ECC and Bayesian learning
- approach the theoretical limit in some cases

**let’s compare!**
Different ECCs on 3-label Powerset (scene data set w/ \( L = 6 \))

- learner: 3-label powerset with Random Forests
- REP + 3-label powerset \( \approx \) RAKEL

Comparing to RAKEL (on most of data sets),
- HAMR: **better 0/1 loss**, similar Hamming loss
- BCH: **even better 0/1 loss**, pay for Hamming loss
Different ECCs on Binary Relevance (scene data set with \( L = 6 \))

- Binary Relevance: simply 1-label powerset
- REP + Binary Relevance \( \approx \) Binary Relevance (with aggregation)

### 0/1 loss

Comparing to BR (on most of data sets),

- BCH/HAMR + BR: **better 0/1 loss, better Hamming loss**
reduction to larger multi-label classification
encode via error-correcting code and capture label combinations (parity bits)
effective decoding (error-correcting)
simple theoretical guarantee + good practical performance
  to improve RA$kEL$, replace REP by
    HAMR $\Rightarrow$ lower 0/1 loss, similar Hamming loss
    BCH $\Rightarrow$ even lower 0/1 loss, but higher Hamming loss
  to improve Binary Relevance, use
    HAMR or BCH $\Rightarrow$ lower 0/1 loss, lower Hamming loss
Conclusion

1. **Coding/Geometric View** of Multi-label Classification
   —useful in linking to Information Theory & visualizing

2. **Compression Coding**
   —condense for efficiency: better approach PLST

3. **Error-correction Coding**
   —expand for accuracy: better code HAMR or BCH

More......

- more geometric explanations  (Tai & Lin, NC Journal 2012)
- beyond standard ECC-decoding  (Ferng, NTU Thesis 2012)
- improved PLST  (Chen, NTU Thesis 2012)
- dynamic instead of static coding  (...), combine ML-ECC & PLST  (...)

Thank you. Questions?