~~Label Space Coding for Multi-label Classification~~

Mathematical Machine Learning for Modern Artificial Intelligence

Hsuan-Tien Lin

National Taiwan University

7th ~~6th~~ ~~3rd~~ TWSIAM Annual Meeting, 05/25/2019

From Intelligence to Artificial Intelligence

intelligence: thinking and acting smartly

- humanly
- rationally

artificial intelligence: **computers** thinking and acting smartly

- humanly
- rationally

humanly $\approx$ **smartly** $\approx$ rationally
—are humans rational? :-)

Traditional vs. Modern [My] Definition of AI

Traditional Definition

humanly $\approx$ intelligently $\approx$ rationally

My Definition

intelligently $\approx$ easily

is your smart phone 'smart'? :-)

modern artificial intelligence
$=$ **application** intelligence

Examples of Application Intelligence

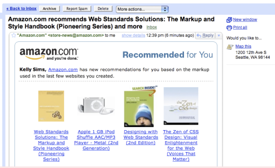
Siri

By Bernard Goldbach [CC BY 2.0]

iRobot

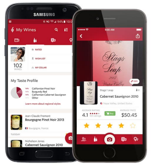
By Yuan-Chou Lo [CC BY-NC-ND 2.0]

Amazon Recommendations



By Kelly Sims [CC BY 2.0]

Vivino



from `nordic.businessinsider.com`

Machine Learning and AI

Easy-to-Use

Acting Humanly

Acting Rationally

Machine Learning

machine learning: core behind modern (data-driven) AI

ML Connects Big Data and AI

From Big Data to Artificial Intelligence

big data $\longrightarrow$ $\boxed{\text{ML}}$ $\longrightarrow$ artificial intelligence

ingredient

$\longrightarrow$

tools/steps

$\longrightarrow$

dish

(Photos Licensed under CC BY 2.0 from Andrea Goh on Flickr)

"cooking" needs many possible
tools & procedures

Bigger Data Towards Better AI

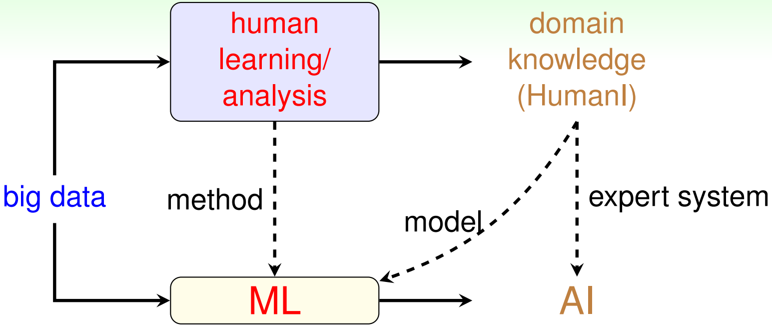
best route by shortest path

best route by current traffic

best route by predicted travel time

big data **can** make machine look smarter

ML for Modern AI

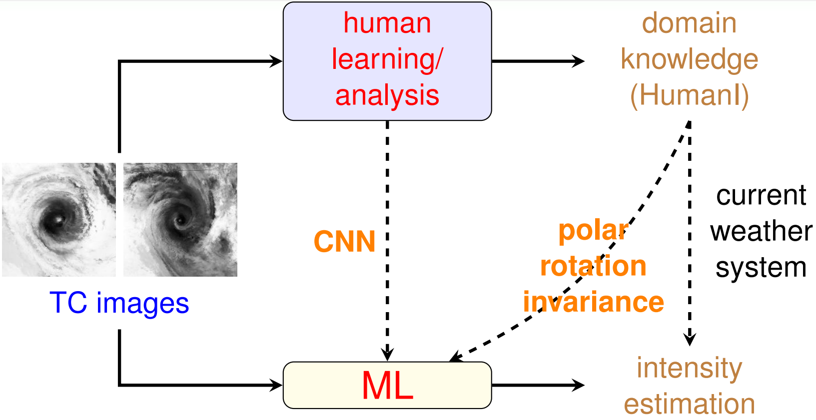


- human sometimes **faster learner** on **initial (smaller) data**
- industry: **black plum is as sweet as white**

often important to leverage human learning, especially **in the beginning**

Application: Tropical Cyclone Intensity Estimation

meteorologists can ‘feel’ & estimate TC intensity from image



better than current system & ‘**trial-ready**’

(Chen et al., KDD 2018)

(Chen et al., Weather & Forecasting 2019)

Cost-Sensitive Multiclass Classification

Patient Status Prediction

?

H7N9-infected

cold-infected

healthy

error measure = society cost

actual \ predicted	H7N9	cold	healthy
H7N9	0	1000	**100000**
cold	100	0	3000
healthy	100	30	0

- H7N9 mis-predicted as healthy: **very high cost**
- cold mis-predicted as healthy: high cost
- cold correctly predicted as cold: no cost

human doctors consider costs of decision;
how about computer-aided diagnosis?

Setup: Cost-Sensitive Classification

Given

N classification examples (input $\mathbf{x}_n$, label y_n) $\in \mathcal{X} \times \{1, 2, \ldots, K\}$

$\boxed{\text{and a 'proper' cost matrix } C \in \mathbb{R}^{K \times K}}$

Goal

a classifier $g(\mathbf{x})$ that $\boxed{\text{pays a small cost } C(y, g(\mathbf{x}))}$ on future **unseen** example $(\mathbf{x}, y)$

cost-sensitive classification:

more **application-realistic**
than traditional classification

Key Idea: Cost Estimator (Tu and Lin, ICML 2010)

Goal

a classifier $g(\mathbf{x})$ that $\boxed{\text{pays a small cost } C(y, g(\mathbf{x}))}$ on future **unseen** example $(\mathbf{x}, y)$

consider expected conditional costs $\mathbf{c}_{\mathbf{x}}[k] = \sum_{y=1}^{K} C(y, k)P(y|\mathbf{x})$

if $\mathbf{c}_{\mathbf{x}}$ known

optimal

$g^*(\mathbf{x}) = \text{argmin}_{1 \le k \le K} \, \mathbf{c}_{\mathbf{x}}[k]$

if $r_k(\mathbf{x}) \approx \mathbf{c}_{\mathbf{x}}[k]$ well

approximately good

$g_r(\mathbf{x}) = \text{argmin}_{1 \le k \le K} \, r_k(\mathbf{x})$

how to get cost estimator r_k? **regression**

Cost Estimator by Per-class Regression

Given

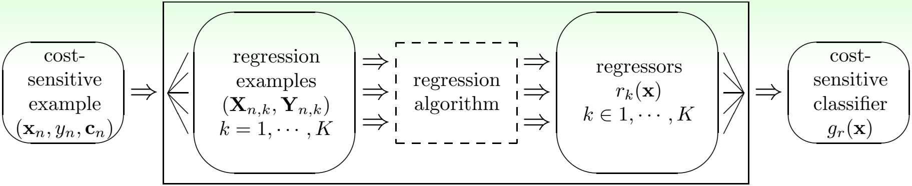
N examples, each (input $\mathbf{x}_n$, label y_n) $\in \mathcal{X} \times \{1, 2, \ldots, K\}$

- take $\mathbf{c}_n$ as y_n-th row of C: $\mathbf{c}_n[k] = C(y_n, k)$

$$
\underbrace{\begin{array}{cc} \text{input} & \mathbf{c}_n[1] \\ \mathbf{x}_1 & 0, \\ \mathbf{x}_2 & 1, \\ \cdots & \\ \mathbf{x}_N & 6, \end{array}}_{r_1}
\;\Bigg\|\;
\underbrace{\begin{array}{cc} \text{input} & \mathbf{c}_n[2] \\ \mathbf{x}_1 & 2, \\ \mathbf{x}_2 & 3, \\ & \\ \mathbf{x}_N & 1, \end{array}}_{r_2}
\;\Bigg\|\; \ldots \;\Bigg\|\;
\underbrace{\begin{array}{cc} \text{input} & \mathbf{c}_n[K] \\ \mathbf{x}_1 & 1 \\ \mathbf{x}_2 & 5 \\ & \\ \mathbf{x}_N & 0 \end{array}}_{r_K}
$$

want: $r_k(\mathbf{x}) \approx \mathbf{c}_{\mathbf{x}}[k]$ for all future $\mathbf{x}$ and k

The Reduction Framework



1. transform classification examples $(\mathbf{x}_n, y_n)$ to regression examples $(\mathbf{x}_{n,k}, Y_{n,k}) = (\mathbf{x}_n, C(y_n, k))$
2. use your favorite algorithm on the regression examples and get estimators $r_k(\mathbf{x})$
3. for each new input $\mathbf{x}$, predict its class using $g_r(\mathbf{x}) = \text{argmin}_{1 \le k \le K}\, r_k(\mathbf{x})$

the reduction-to-regression framework:
systematic & easy to implement

A Simple Theoretical Guarantee

$$g_r(\mathbf{x}) = \operatorname*{argmin}_{1 \le k \le K} r_k(\mathbf{x})$$

Theorem (Absolute Loss Bound)

For any set of estimators (cost estimators) $\{r_k\}_{k=1}^{K}$ *and for any tuple* $(\mathbf{x}, y, \mathbf{c})$ *with* $\mathbf{c}[y] = 0 = \min_{1 \le k \le K} \mathbf{c}[k]$,

$$\mathbf{c}[g_r(\mathbf{x})] \le \sum_{k=1}^{K} \Big| r_k(\mathbf{x}) - \mathbf{c}[k] \Big|.$$

low-cost classifier $\Longleftarrow$ accurate estimator

Our Contributions

In 2010 (Tu and Lin, ICML 2010)

- **tighten** the simple guarantee (**+math**)
- **propose** loss function (**+math**) from tighter bound
- **derive** SVM-based model (**+math**) from loss function

—eventually reaching **superior experimental results**

Six Years Later (Chung et al., IJCAI 2016)

- **propose smoother** loss function (**+math**) from tighter bound
- **derive** world's first cost-sensitive deep learning model (**+math**) from loss function

—eventually reaching **even superior experimental results**

why are people **not**
using those **cool ML works for their AI? :-)**

Issue 1: Where Do Costs Come From?

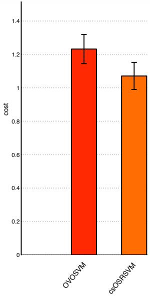
A Real Medical Application: Classifying Bacteria

- by human doctors: **different treatments** $\Longleftrightarrow$ serious costs
- cost matrix averaged from two doctors:

	Ab	Ecoli	HI	KP	LM	Nm	Psa	Spn	Sa	GBS
Ab	0	1	10	7	9	9	5	8	9	1
Ecoli	3	0	10	8	10	10	5	10	10	2
HI	10	10	0	3	2	2	10	1	2	10
KP	7	7	3	0	4	4	6	3	3	8
LM	8	8	2	4	0	5	8	2	1	8
Nm	3	10	9	8	6	0	8	3	6	7
Psa	7	8	10	9	9	7	0	8	9	5
Spn	6	10	7	7	4	4	9	0	4	7
Sa	7	10	6	5	1	3	9	2	0	7
GBS	2	5	10	9	8	6	5	6	8	0

issue 2: is cost-sensitive classification **really useful**?

Cost-Sensitive vs. Traditional on Bacteria Data

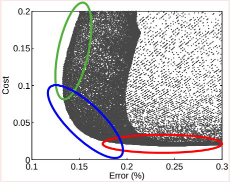


(Jan et al., BIBM 2011)

cost-sensitive better than **traditional**;
but why are people **still not**
using those cool ML works for their AI? :-)

Issue 3: Error Rate of Cost-Sensitive Classifiers

The Problem



- cost-sensitive classifier: low cost but high error rate
- traditional classifier: low error rate but high cost
- how can we get the blue classifiers?: low error rate and low cost —**math++** on **multi-objective** optimization (Jan et al., KDD 2012)

now, **are people using those cool ML works for their AI? :-)**

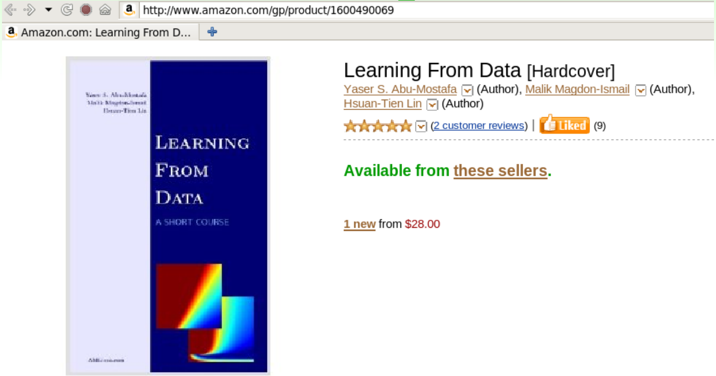
Lessons Learned from Research on Cost-Sensitive Multiclass Classification

			
?	H7N9-infected	cold-infected	healthy

1. more realistic (generic) in academia
   $\neq$ **more realistic (feasible) in application**
   e.g. the 'cost' of **inputting a cost matrix? :-)**
2. **cross-domain collaboration** important
   e.g. getting the 'cost matrix' from **domain experts**
3. not easy to win **human trust**
   —humans are somewhat **multi-objective**
4. many battlefields for **math** towards application intelligence
   e.g. abstraction of **goals and needs**

Label Space Coding for Multilabel Classification

What **Tags**?



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

a **multilabel** classification problem:
tagging input to multiple categories

Binary Relevance: Multilabel Classification via Yes/No

Binary Classification

{yes, no}

multilabel w/ L classes: L Y/N 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:
  transformation 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)
  - **unbalanced**—few yes, many no

Binary Relevance: simple (& good) benchmark with known disadvantages

From Label-set to Coding View

	label set	apple	orange	strawberry	**binary code**
	{o}	0 (N)	1 (Y)	0 (N)	[0, 1, 0]
	{a, o}	1 (Y)	1 (Y)	0 (N)	[1, 1, 0]
	{a, s}	1 (Y)	0 (N)	1 (Y)	[1, 0, 1]
	{o}	0 (N)	1 (Y)	0 (N)	[0, 1, 0]
	{}	0 (N)	0 (N)	0 (N)	[0, 0, 0]

subset of $2^{\{1,2,\cdots,L\}}$ $\Leftrightarrow$ **length-L binary code**

A NeurIPS 2009 Approach: Compressive Sensing

General Compressive Sensing

sparse (many 0) binary vectors $\mathbf{y} \in \{0, 1\}^L$ can be **robustly compressed** by projecting to $M \ll L$ basis vectors $\{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_M\}$

Comp. Sensing for Multilabel Classification (Hsu et al., NeurIPS 2009)

1. **compress**: encode original data by **compressive sensing**
2. **learn**: get regression function from compressed data
3. **decode**: decode regression predictions to sparse vector by **compressive sensing**

Compressive Sensing: seemly strong competitor **from related theoretical analysis**

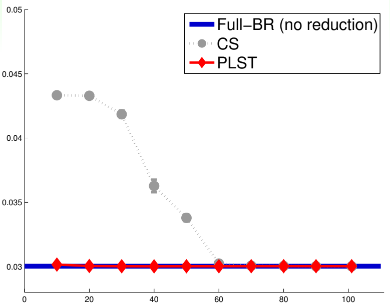
Our Proposed Approach: Compressive Sensing ⇒ PCA

Principal Label Space Transformation (PLST), i.e. PCA for Multilabel Classification (Tai and Lin, NC Journal 2012)

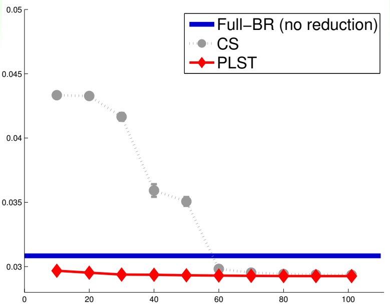
1. **compress**: encode original data by **PCA**
2. **learn**: get regression function from compressed data
3. **decode**: decode regression predictions to label vector by **reverse PCA + quantization**

does PLST perform better than CS?

Hamming Loss Comparison: PLST vs. CS



mediamill (Linear Regression)



mediamill (Decision Tree)

- **PLST** better than CS: faster, **better** performance
- similar findings across **data sets** and **regression algorithms**

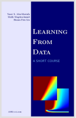
Why? CS creates
harder-to-learn regression tasks

Our Works Continued from PLST

1. **Compression** Coding (Tai & Lin, NC Journal 2012 with 216 citations)
   —**condense** for efficiency: better (than CS) approach PLST
   — key tool: PCA from Statistics/Signal Processing
2. **Learnable-Compression** Coding (Chen & Lin, NeurIPS 2012 with 157 citations)
   —**condense learnably** for **better** efficiency: better (than PLST) approach CPLST
   — key tool: Ridge Regression from Statistics (+ PCA)
3. **Cost-Sensitive** Coding (Huang & Lin, ECML Journal Track 2017)
   —**condense cost-sensitively** towards application needs: better (than CPLST) approach CLEMS
   — key tool: Multidimensional Scaling from Statistics

cannot thank **statisticans**
enough for those tools!

Lessons Learned from Label Space Coding for Multilabel Classification



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

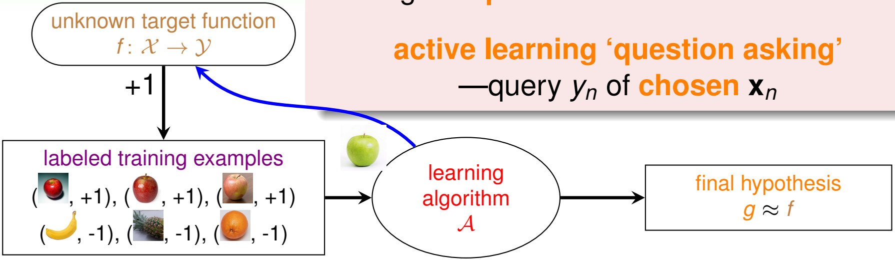
1. Is Statistics the same as ML? Is Statistics the same as AI?
   - **does it really matter?**
   - Modern AI should embrace **every useful tool from every field** & any necessary **math**
2. 'application intelligence' tools **not necessarily most sophisticated ones**
   e.g. PCA possibly more useful than CS for label space coding
3. more-cited paper $\neq$ more-useful AI solution
   —citation count **not the only impact measure**
4. **are people using those cool ML works for their AI?**
   —we wish!

Active Learning by Learning

Active Learning: Learning by 'Asking'

labeling is **expensive**:

active learning 'question asking'
—query y_n of **chosen** $\mathbf{x}_n$



active: improve hypothesis with fewer labels
(hopefully) by asking questions **strategically**

Pool-Based Active Learning Problem

Given

- labeled pool $\mathcal{D}_l = \left\{ (\text{feature } \mathbf{x}_n$  $, \textbf{label } y_n \text{ (e.g. IsApple?)}) \right\}_{n=1}^{N}$
- unlabeled pool $\mathcal{D}_u = \left\{ \tilde{\mathbf{x}}_s \right\}_{s=1}^{S}$

Goal

design an algorithm that iteratively

1. **strategically query** some $\tilde{\mathbf{x}}_s$  to get associated $\tilde{y}_s$
2. move $(\tilde{\mathbf{x}}_s, \tilde{y}_s)$ from $\mathcal{D}_u$ to $\mathcal{D}_l$
3. learn **classifier** $g^{(t)}$ from $\mathcal{D}_l$

and improve **test accuracy of** $g^{(t)}$ w.r.t **#queries**

how to **query strategically**?

How to Query Strategically?

Strategy 1

ask **most confused** question

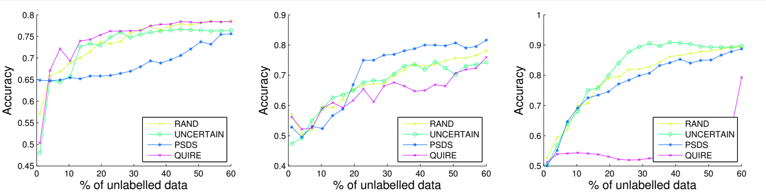
Strategy 2

ask **most frequent** question

Strategy 3

ask **most debateful** question

- **choosing** one single strategy is **non-trivial**:



application intelligence: how to **choose strategy smartly**?

Idea: Trial-and-Reward Like Human

when do humans **trial**-and-**reward**?
gambling

K strategies:

$\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_K$

try one strategy

"goodness" of strategy as **reward**

K bandit machines:

$\mathcal{B}_1, \mathcal{B}_2, \cdots, \mathcal{B}_K$

try one bandit machine

"luckiness" of machine as **reward**

intelligent choice of strategy
$\Longrightarrow$ intelligent choice of **bandit machine**

Active Learning by Learning (Hsu and Lin, AAAI 2015)

K strategies:
$\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_K$

→ **try** one strategy

↓

"goodness" of strategy as **reward**

← (back to K strategies)

Given: K existing active learning strategies

for $t = 1, 2, \ldots, T$

1. let some bandit model **decide strategy $\mathcal{A}_k$ to try**
2. **query the $\tilde{\mathbf{x}}_s$** suggested by $\mathcal{A}_k$, and compute $g^{(t)}$
3. evaluate **goodness of $g^{(t)}$** as **reward** of **trial** to update model

only remaining problem: **what reward**?

Design of Reward

ideal reward after updating classifier $g^{(t)}$ by the query $(\mathbf{x}_{n_t}, y_{n_t})$:

accuracy of $g^{(t)}$ on **test set** $\{(\mathbf{x}'_m, y'_m)\}_{m=1}^{M}$

—test accuracy **infeasible** in practice because labeling **expensive**

more feasible reward: training accuracy on the fly

accuracy of $g^{(t)}$ on **labeled pool** $\{(\mathbf{x}_{n_\tau}, y_{n_\tau})\}_{\tau=1}^{t}$

—but **biased** towards **easier** queries

weighted training accuracy as a better reward:

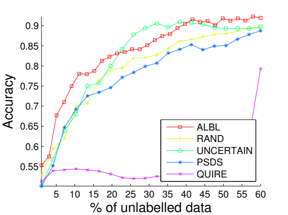
acc. of $g^{(t)}$ on inv.-prob. weighted **labeled pool** $\left\{\left(\mathbf{x}_{n_\tau}, y_{n_\tau}, \frac{1}{p_\tau}\right)\right\}_{\tau=1}^{t}$

—'bias correction' from querying probability within bandit model

Active Learning by Learning (ALBL):
bandit + **weighted training acc. as reward**

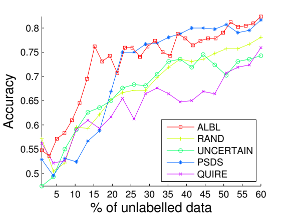
Comparison with Single Strategies

UNCERTAIN Best



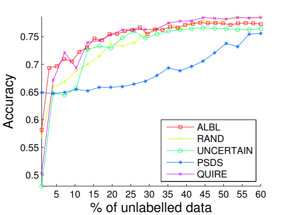
vehicle

PSDS Best



sonar

QUIRE Best



diabetes

- **no single best strategy** for every data set
  —choosing needed
- proposed **ALBL** consistently **matches the best**
  —similar findings across other data sets

ALBL: effective in **making intelligent choices**

Discussion for Statisticians

weighted training accuracy $\frac{1}{t}\sum_{\tau=1}^{t} \frac{1}{p_\tau} [\![y_{n_\tau} = g^{(t)}(\mathbf{x}_{n_\tau})]\!]$ as reward

- is reward unbiased estimator of test performance?
  no for learned $g^{(t)}$ (yes for fixed g)
- is reward fixed before playing?
  no because $g^{(t)}$ learned from $(\mathbf{x}_{n_t}, y_{n_t})$
- is reward independent of each other?
  no because past history all in reward

—ALBL: tools from statistics + **wild/unintended usage**

'application intelligence' outcome:
open-source tool released
(`https://github.com/ntucllab/libact`)

Lessons Learned from Research on Active Learning by Learning

by DFID - UK Department for International Development; licensed under CC BY-SA 2.0 via Wikimedia Commons

1. **scalability bottleneck** of 'application intelligence': **choice** of methods/models/parameter/...
2. think outside of the **math** box: 'unintended' usage may be **good enough**
3. important to be **brave** yet **patient**
   —**idea: 2012**
   —paper (Hsu and Lin, AAAI 2015); software (Yang et al., 2017)

Summary

- ML for (Modern) AI:
  tools + human knowledge
  $\Rightarrow$ **easy-to-use application intelligence**
- ML Research for Modern AI:
  need to be **more open-minded**
  —in methodology, in collaboration, in KPI
- **Math** in ML Research for Modern AI:
  —**new setup/need/goal** & wider usage of tools

Thank you! Questions?