Hands-on Learning to Search for Structured Joint Prediction

Kai-Wei Chang @UIUC->MSR-NE

Hal Daume, He He, Sudha Rao @UMaryland

John Langford @MSR-NYC

http://tinyurl.com/naacl15l2s

NAACL, May 31
The Problem: Joint Prediction

How? Other answers
The Problem: Joint Prediction

How? Other answers

1. Each prediction is independent.
The Problem: Joint Prediction

How? Other answers

1. Each prediction is independent.
The Problem: Joint Prediction

How? Other answers

1. Each prediction is independent.
3. Assume tractable graphical model, optimize.
The Problem: Joint Prediction

How? Other answers
1. Each prediction is independent.
3. Assume tractable graphical model, optimize.
What makes a good solution?

1. Programming complexity.
What makes a good solution?

1. **Programming complexity.** Most complex problems addressed independently—too complex to do otherwise.
What makes a good solution?

1. **Programming complexity.** Most complex problems addressed independently—too complex to do otherwise.

2. **Prediction accuracy.** It had better work well.
What makes a good solution?

1. **Programming complexity.** Most complex problems addressed independently—too complex to do otherwise.

2. **Prediction accuracy.** It had better work well.

3. **Train speed.** Debug/development productivity + maximum data input.
What makes a good solution?

1. **Programming complexity.** Most complex problems addressed independently—too complex to do otherwise.

2. **Prediction accuracy.** It had better work well.

3. **Train speed.** Debug/development productivity + maximum data input.

4. **Test speed.** Application efficiency
A program complexity comparison

```
1 10 100 1000
CRFSGD CRF++ S-SVM Search
```

lines of code for POS
POS Tagging (tuned hps)

Accuracy (per word)

Training time (minutes)

OAA
L2S
L2S (ft)
CRFsgd
CRF++
StrPerc
StrSVM
StrSVM2
The Plan

Part 1

1. Machine Learning + Vowpal Wabbit Background
   1. Input format
   2. One-against-all Multiclass
   3. Regression
   2. Joint prediction by Imitation Learning
   3. Joint Prediction by Learning to Search

Part 2: Hands-on.
Let’s solve Named Entity Recognition.
wget http://bit.ly/1FVkLEK
Part of Speech Tagging

wget http://bit.ly/1FVkLEK
unzip 1FVkLEK
Part of Speech Tagging

wget http://bit.ly/1FVkLEK
unzip 1FVkLEK
less wsj.train.vw

1 w Despite
2 w continuing
3 w problems
1 w in
4 w its
5 w newsprint
5 w business

In general:
label | Namespace Feature
An approach: One-Against-All (OAA)

Create $k$ binary problems, one per class. For class $i$ predict “Is the label $i$ or not?”

\[(x, y) \mapsto \begin{cases} (x, 1(y = 1)) \\ (x, 1(y = 2)) \\ \vdots \\ (x, 1(y = k)) \end{cases}\]

Then reduce to regression.
Key ideas:

1. Hashing: avoid dictionaries for simplicity/speed.
2. Online Learning: quick optimization.
3. Progressive Validation: test one ahead of train for unbiased perf.
Key ideas:

1. Hashing: avoid dictionaries for simplicity/speed.
2. Online Learning: quick optimization.
3. Progressive Validation: test one ahead of train for unbiased perf.
Cost-sensitive multiclass classification

Distribution $D$ over $X \times [0, 1]^k$, where a vector in $[0, 1]^k$ specifies the cost of each of the $k$ choices.

Find a classifier $h : X \rightarrow \{1, \ldots, k\}$ minimizing the expected cost

$$\text{cost}(c, D) = E_{(x,c)\sim D}[c_{h(x)}].$$
Cost-sensitive multiclass classification

Distribution $D$ over $X \times [0, 1]^k$, where a vector in $[0, 1]^k$ specifies the cost of each of the $k$ choices.

Find a classifier $h : X \rightarrow \{1, \ldots, k\}$ minimizing the expected cost

$$\text{cost}(c, D) = \mathbb{E}_{(x, c) \sim D}[c_h(x)].$$

1. Is this packet \{normal, error, attack\}?
2. A key primitive for learning to search.
Use in VW

Label information via sparse vector.
A test example:
|Namespace Feature
A test example with only classes 1,2,4 valid:
1: 2: 4: |Namespace Feature
A training example with only classes 1,2,4 valid:
1:0.4 2:3.1 4:2.2 |Namespace Feature
Use in VW

Label information via sparse vector.
A test example:

| Namespace Feature

A test example with only classes 1, 2, 4 valid:
1: 2: 4:  | Namespace Feature

A training example with only classes 1, 2, 4 valid:
1:0.4 2:3.1 4:2.2  | Namespace Feature

Methods:
- csoaa $k$ cost-sensitive OAA prediction. $O(k)$ time.
- csoaa_ldf Label-dependent features OAA.
- wap_ldf LDF Weighted-all-pairs.
tar -xvzf 1EvYlm9
zless VW_raw/rcv1.train.raw.txt.gz
The reduction to regression

tar -xvzf 1EvYlm9
zless VW_raw/rcv1.train.raw.txt.gz
1 | tuesday year million short compan ...
-1 | econom stock rate month year invest ...
...
The reduction to regression

tar -xvzf 1EvYlm9
zless VW_raw/rcv1.train.raw.txt.gz
1 | tuesday year million short compan ...
-1 | econom stock rate month year invest ...
...
vw -c rcv1.train.raw.txt -b 22 --ngram 2
--skips 4 -l 0.25 --binary
generates good solution.
Features: a vector $x \in \mathbb{R}^n$
Label: $y \in \mathbb{R}$
Goal: Learn $w \in \mathbb{R}^n$ such that $\hat{y}_w(x) = \sum_i w_i x_i$ is close to $y$. 
Online Linear Learning

Start with $\forall i : \ w_i = 0$

Repeatedly:

1. Get features $x \in \mathbb{R}^n$.
2. Make linear prediction $\hat{y}_w(x) = \sum_i w_i x_i$.
3. Observe label $y \in \mathbb{R}$.
4. Update weights so $\hat{y}_w(x)$ is closer to $y$.
   Example: $w_i \leftarrow w_i + \eta(y - \hat{y})x_i$. 
Repeated squared loss updates

Squared Loss
Repeated update

loss

prediction when y=1
The Plan

1. Part 1
   1. Machine Learning + Vowpal Wabbit Background
   2. Joint prediction by Imitation Learning
   3. Joint Prediction by Learning to Search

   Let’s solve Named Entity Recognition.