Online Learning for Phrase Recognition

Xavier Carreras and Lluís Màrquez

July 2003
Phrase Recognition
Phrase Recognition
Phrase Recognition

Online Learning for Phrase Recognition
Phrase Recognition

• ((• •) (•) •)
A solution is a coherent set of (embedded) phrases.
Phrase Overlapping
Phrase Overlapping
Phrase Overlapping

( • ( • • ) • ) • •
Some Solution Candidates

(((● (●)) ● (●) )
Some Solution Candidates

(((●) (●)) ● (●) )

((●) (●) (●) (●) (●))
Some Solution Candidates

(((● (●)) ● (●) ))

((● (●) (●) (●))

(((●) (●) ● ) (●) )

(((●) (●) (●) (●)))
Some Solution Candidates

(((●) (●)) ● (●))

((●) (●) (●) (●))

((((●) (●) ●) (●))

(● (((●) (●)) ●))
Phrase Score

We learn to score phrases. \( \forall k \in \mathcal{K} : \)

\[ \text{score}_k(s, e) \rightarrow \mathbb{R} \]

Given the score of \((s, e)\):

- The sign tells whether \((s, e)\) is a \(k\)-phrase or not.
- The magnitude codifies the confidence of the decision.
Phrase Recognition Model

$\mathcal{Y}$: solution space, i.e. set of all coherent phrase sets.

$$\text{PhRec}(x) = \arg \max_{y \in \mathcal{Y}} \sum_{(s,e)_k \in y} \text{score}_k(s,e)$$

- Sequential case: $O(n^2)$ Dynamic Prog. search
- Hierarchical case: $O(n^3)$ Dynamic Prog. search
Phrase Recognition Model: Start-End Candidates + Phrase Scoring

\( \mathcal{Y} \): solution space, i.e. set of all coherent phrase sets.
\( \mathcal{Y}_{SE} \): practical solution space, filtered at word level.

\[
\text{PhRec}(x) = \arg \max_{y \in \mathcal{Y}_{SE}} \sum_{(s,e)_k \in y} \text{score}_k(s, e)
\]

\[
\mathcal{Y}_{SE} = \{ y \in \mathcal{Y} \mid \forall (s, e)_k \in y \quad \text{start}_k(s) \land \text{end}_k(e) \}\}
Online Learning for Phrase Recognition
Learning Challenges

• Learn all functions \((\text{start}_k, \text{end}_k, \text{score}_k)\) so as to maximize the \(F_1\) measure on the recognition of phrases.

• Start-End:
  ★ As filters, rather than default classifiers.
  ★ They define the input space to the \text{score} functions

• Score functions:
  ★ The negative space is too big \(\sim O(n^2)\).
  ★ We need to know about Start-End behavior.
  ★ As rankers, rather than default classifiers.
Score as a Classifier
Score as a Classifier

```
(( ( ) ) )
```

```
+ 0 -
```
Score as a Ranker

(( • (•)) (• (•) • ))
Score as a Ranker

((( ⋅ (●)) (● (●) ⋅ ))
Score as a Ranker

( ( • ( • ) ) ( • ( • ) • ) )

Online Learning for Phrase Recognition
Score as a Ranker

((● (●)) (● (●) ● ))

+ 0
Ranking Motivation

(The cat) (eats) (fresh fish).
Ranking Motivation

(The cat) (eats) (fresh fish) . (The cat) (eats) (fish) .
Ranking Motivation

(The cat) (eats) (fresh fish) .  (The cat) (eats) (fish) .
Ranking Motivation

(The cat) (eats) (fresh fish) . (The cat) (eats) (fish) .

Online Learning for Phrase Recognition
Perceptron Learning Algorithm

- Input: \{ (x^1, y^1), \ldots, (x^m, y^m) \}, \( x^i \) are sentences, \( y^i \) are solutions
- Define: \( W = \{ w_S, w_E \} \cup \{ w_k | k \in K \} \).
- Initialize: \( \forall w \in W \ w = 0 \);
- for \( t = 1 \ldots T \), for \( i = 1 \ldots m \):
  1. \( \hat{y} = \text{PhRec}_W(x^i) \)
  2. learning_feedback(\( W, x^i, y^i, \hat{y} \))
- Output: the vectors in \( W \).
Learning Feedback (I)

- Phrases correctly identified: \( \forall (s, e)_k \in y^* \cap \hat{y} \):
  - Do nothing, since they are correct.

- Missed phrases: \( \forall (s, e)_k \in y^* \setminus \hat{y} \):
  1. Update misclassified boundary words:
     - if \( (w_S \cdot \Phi_w(x_s) \leq 0) \) then \( w_S = w_S + \Phi_w(x_s) \)
     - if \( (w_E \cdot \Phi_w(x_e) \leq 0) \) then \( w_E = w_E + \Phi_w(x_e) \)
  2. Update score function, if applied:
     - if \( (w_S \cdot \Phi_w(x_s) > 0 \land w_E \cdot \Phi_w(x_e) > 0) \) then
       \( w_k = w_k + \Phi_p(s, e) \)
Learning Feedback (II)

• Over-predicted phrases: \( \forall (s, e)_k \in \hat{y} \setminus y^* : \)

1. Update score function:
   \[ w_k = w_k - \Phi_p(s, e) \]
2. Update words misclassified as S or E:
   if \( \text{goldS}(s) = 0 \) then \( w_S = w_S - \Phi_w(x_s) \)
   if \( \text{goldE}(e) = 0 \) then \( w_E = w_E - \Phi_w(x_e) \)
Experiments on Clause Identification
Experiments on Clause Identification

![Graph showing precision and recall for different models over epochs.]

- **Legend:**
  - Precision: Rec-VP, CI-VP, CI-SVM
  - Recall: Rec-VP, CI-VP, CI-SVM

- **Y-axis:** Precision/Recall on Start words
- **X-axis:** Number of Epochs
# Experiments on Clause Identification

<table>
<thead>
<tr>
<th></th>
<th>development</th>
<th></th>
<th></th>
<th>test</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>T</td>
<td>prec.</td>
<td>recall</td>
<td>$F_{\beta=1}$</td>
<td>prec.</td>
<td>recall</td>
</tr>
<tr>
<td>Cl-VP</td>
<td>7</td>
<td>87.24</td>
<td>72.83</td>
<td>79.39</td>
<td>85.19</td>
<td>65.49</td>
</tr>
<tr>
<td>Cl-SVM</td>
<td>–</td>
<td>87.70</td>
<td>75.38</td>
<td>81.07</td>
<td>86.94</td>
<td>68.40</td>
</tr>
<tr>
<td>Rec-VP</td>
<td>11</td>
<td>89.80</td>
<td><strong>84.05</strong></td>
<td>86.83</td>
<td>87.99</td>
<td><strong>75.29</strong></td>
</tr>
<tr>
<td>AdaBoost (1)</td>
<td>–</td>
<td><strong>92.53</strong></td>
<td>82.48</td>
<td><strong>87.22</strong></td>
<td><strong>90.18</strong></td>
<td>72.59</td>
</tr>
<tr>
<td>AdaBoost (2)</td>
<td>–</td>
<td>87.18</td>
<td>82.48</td>
<td>84.77</td>
<td>84.82</td>
<td>73.28</td>
</tr>
</tbody>
</table>

(1) (Carreras, Màrquez, Punyakanok and Roth, ECML’02)
(2) (Carreras and Màrquez, CoNLL’01)