Robust Information Extraction with Perceptrons

Mihai Surdeanu, Massimiliano Ciaramita

Technical University of Catalonia, Yahoo! Research Barcelona
surdeanu@lsi.upc.edu, massi@yahoo-inc.com

March 19, 2007

Outline

Introduction

Architecture

EMD as Sequence Tagging

Ambiguity Detection for EMD

RMD with Perceptron with Dynamic Uneven Margins

Experiments

Conclusions

Introduction

- How far can you get with a “practical” IE system?
  - Small development time: model everything using Machine Learning and simple features sets.
  - Small training/testing times: use online learning.
  - Robustness: use only NLP preprocessing tools that work well on any corpus: part of speech (POS) tagging and chunking.
- Novel issues:
  - All learning tasks modeled using variants of the Perceptron Algorithm (PA). For RMD, we propose a new large-margin PA tailored for class-unbalanced data → performed better than SVM and PA.
  - Novel architecture to mitigate errors in early stages (entity classification) → let ambiguities trickle through the following learning components and solve them only at the end using approximated inference.
- Participated in the English evaluation for Entity Mention Detection (EMD) and Relation Mention Detection (RMD) → obtained competitive results.

Outline

Introduction

Architecture

EMD as Sequence Tagging

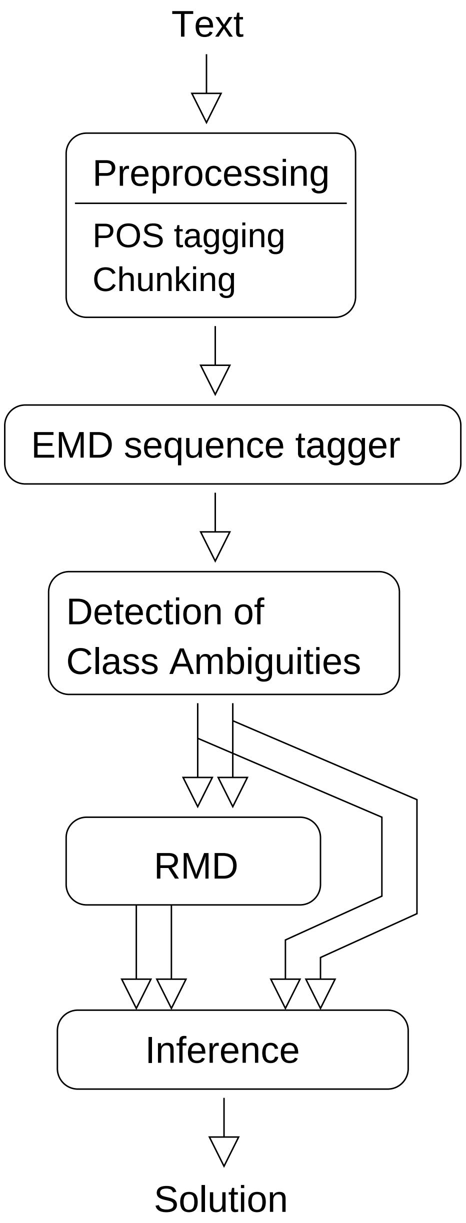
Ambiguity Detection for EMD

RMD with Perceptron with Dynamic Uneven Margins

Experiments

Conclusions

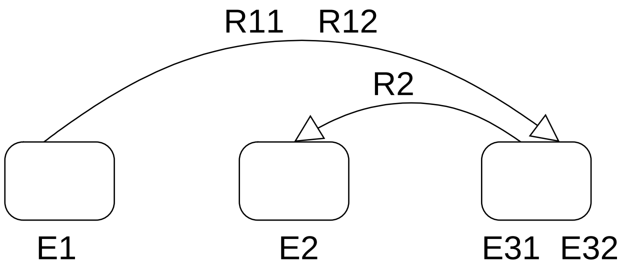
System Architecture



- POS tagging – TnT, chunking – Yamcha.
- EMD: sequence tagger (BIO) using the PA for structure learning. We model entity type + subtype jointly, e.g., `B-FAC-Plant` marks the beginning of an entity mention of type FAC and subtype `Plant`.
- Detection of class ambiguities: if EMD not confident enough in entity classification → let several classes pass through to RMD.
- RMD: classifies every possible pair of entity mentions. Very unbalanced data: ratio of – to + examples more than 13 to 1 → new large-margin PA tailored for class-unbalanced scenarios.
- Inference: combines all possible outputs into a single consistent solution.

Inference

- Candidate generation:



  The following candidates are generated for the above sentence: {R11(E1, E31), R2(E31, E2)}, {R11(E1, E32), R2(E32, E2)}, {R12(E1, E31), R2(E31, E2)}, {R12(E1, E32), R2(E32, E2)}

- Candidate search:
  - Sort candidates in descending order of their confidence:

$$conf(\mathbf{E}, \mathbf{R}) = \lambda_e \sum_{i=1}^{|\mathbf{E}|} p(E_i) + \lambda_r \sum_{i=1}^{|\mathbf{R}|} p(R_i)$$

  - Select the best candidate that satisfies the domain constraints.

Outline

Introduction

Architecture

EMD as Sequence Tagging

Ambiguity Detection for EMD

RMD with Perceptron with Dynamic Uneven Margins

Experiments

Conclusions

EMD as Sequence Tagging

Learner

Hidden Markov Average Perceptron

```
input  : $\mathcal{S} = (\mathbf{x}_i, y_i)^N$; $\mathbf{w}^0 = \vec{0}$
for $t = 1$ to $T$ do
    choose $\mathbf{x}_j$
    compute $\hat{\mathbf{y}} = f_{\mathbf{w_t}}(\mathbf{x_j})$
    if $\hat{\mathbf{y}} \neq \mathbf{y}_j$ then
        $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \Phi(\mathbf{x}_j, \mathbf{y}_j) - \Phi(\mathbf{x}_j, \hat{\mathbf{y}})$
output: $\mathbf{w} = \frac{1}{T} \sum_t \mathbf{w}_t$
```

where:

$$f_{\mathbf{w}}(\mathbf{x}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$$

$$\Phi(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{d} \sum_{j=1}^{|\mathbf{y}|} \phi_i(y_{j-1}, y_j, \mathbf{x})$$

Features

1. **Words:** x_i, x_{i-1}, x_{i-2}, x_{i+1}, x_{i+2};
2. **First sense:** supersense baseline prediction for x_i, fs(x_i);
3. **Combined (1) and (2):** x_i + fs(x_i);
4. **Pos:** pos_i (the POS of x_i), $\text{pos}_{\text{i}-1}$, $\text{pos}_{\text{i}-2}$, $\text{pos}_{\text{i}+1}$, $\text{pos}_{\text{i}+2}$, $\text{pos}_\text{i}[0]$, $\text{pos}_{\text{i}-1}[0]$, $\text{pos}_{\text{i}-2}[0]$, $\text{pos}_{\text{i}+1}[0]$, $\text{pos}_{\text{i}+2}[0]$, pos_comm_i if x_i is a common noun, and pos_prop_i if x_i is a proper noun;
5. **Word shape:** sh(x_i), sh(x_{i-1}), sh(x_{i-2}), sh(x_{i+1}), sh(x_{i+2}). E.g., sh(”Merrill Lynch & Co.”) $=$ Xx $*$ Xx $*$ &Xx.;
6. **Previous label:** entity label y_{i-1}.

Additionally, we add all second-order features of the form $x_i x_j$, i.e., $\Phi^2(\mathbf{x}) = (x_i, x_j)_{(i,j)=(1,1)}^{(d,d)} \rightarrow$ equivalent to a polynomial kernel of degree 2 in a dual model.

Outline

Introduction

Architecture

EMD as Sequence Tagging

Ambiguity Detection for EMD

RMD with Perceptron with Dynamic Uneven Margins

Experiments

Conclusions

Entity Classification as Ambiguity Detection

- Reclassifies all entity mentions in order to detect ambiguities, i.e., entity mentions that are assigned several classes with close probabilities.
- Learner: the averaged PA of Freund and Shapire (1999).
- Raw activations converted to probabilities using the *softmax* function.
- Outputs only classes with probabilities within a certain beam relative to the top class.
- An instance of this classifier is used to detect the entity mention type.

Features

token(entity head word)
WordNet SuperSense of head word
BBN class of head word
tokens(entity inside words)
tokens(entity left context)
tokens(entity right context)
true if entity is known person name
true if entity is known location

where the *token* function extracts the word, lemma, and POS tag of a given token. The *tokens* function constructs unigrams and bigrams of words, lemmas, and POS tags for a given sequence of tokens.

Outline

Introduction

Architecture

EMD as Sequence Tagging

Ambiguity Detection for EMD

RMD with Perceptron with Dynamic Uneven Margins

Experiments

Conclusions

Motivation

- *Maximum or large margin classifiers exhibit good generalization performance* → Perceptron Algorithm with Margins (PAM): learns not only when the prediction is incorrect but also when the model is not confident enough, i.e., the predicted margin $< \tau$.
- *Treat positive and negative examples differently in unbalanced data* → Perceptron Algorithm with Uneven Margin (PAUM): uses two margin parameters, one for positive examples, τ_{+1}, and another for negative examples, τ_{-1} (typically $\tau_{+1} \gg \tau_{-1}$).
- *Tuning PAUM's parameters is both important and difficult*. For example, a value too small for τ_{+1} means that the PAUM acquires too few positive examples and the resulting model fails to generalize well. A value too large for τ_{+1} signifies that the PAUM acquires too many positive examples, with the effect that the model is too eager in predicting positive examples.

Perceptron Algorithm with Dynamic Uneven Margins

```
Perceptron Algorithm with Dynamic Uneven Margins
input  : $\mathcal{Z} = (\mathbf{x}, \mathbf{y}) \in (\mathcal{X} \times \{-1, +1\})^m$,
         $\Gamma_{-1}, \Gamma_{+1} \in \mathbb{R}^+$
         $T, \mathbf{w}_1 = \vec{0}, c_1 = 0, k = 1$
for $j \in \{-1, +1\}$ do
    $\tau_j \leftarrow \Gamma_j$, visited$_j \leftarrow 0$, incorrect$_j \leftarrow 0$
for $t = 1$ to $T$ do
    for $i = 1$ to $m$ do
        (a) compute prediction error rate:
        for $j \in \{-1, +1\}$ do
            if $y_i = j$ then
                visited$_j \leftarrow$ visited$_j + 1$
                if $y_i \langle \mathbf{w}_k, \mathbf{x}_i \rangle \leq 0$ then
                    incorrect$_j \leftarrow$ incorrect$_j + 1$
                err$_j \leftarrow \frac{\text{incorrect}_j}{\text{visited}_j}$
        (b) update vectors:
        if $y_i \langle \mathbf{w}_k, \mathbf{x}_i \rangle \leq \tau_{y_i}$ then
            $\mathbf{w}_{k+1} \leftarrow \mathbf{w}_k + y_i \mathbf{x}_i$
            $c_{k+1} \leftarrow 1$
            $k \leftarrow k + 1$
        else
            $c_k \leftarrow c_k + 1$
        (c) update margins:
        for $j \in \{-1, +1\}$ do
            $\tau_j \leftarrow \text{err}_j \Gamma_j$
output: avg $= \sum_{i=1}^{k} c_i \mathbf{w}_i$
```

Intuition

The margin parameters $\tau_{\pm 1}$ are inversely proportional with the classifier generalization performance for positive/negative examples.

Debug

- Generalization performance estimated based on the current error rate.
- If the classifier has low error rate $\longrightarrow$ converge faster by decreasing $\tau_{\pm 1}$.
- If the classifier has high error rate $\longrightarrow$ continue learning by maintaining large values for $\tau_{\pm 1}$.

Features

tokens(head words of relation arguments)
entities(relation arguments)
tokens(words between relation arguments)
tokens(chunks between relation arguments)
path(chunks between relation arguments)
tokens(words in the relation left context)
tokens(chunks in the relation left context)
tokens(words in the relation right context)
tokens(chunks in the relation right context)

- *tokens* – unigrams and bigrams of words, lemmas, and POS tags for a given sequence of tokens.
- *entities* – extracts the top N predicted entity classes for the two arguments and constructs all possible combinations.
- *path* – constructs two sequences, one of chunk syntactic labels and one of head words for the sequence of chunks between the two arguments.

Outline

Introduction

Architecture

EMD as Sequence Tagging

Ambiguity Detection for EMD

RMD with Perceptron with Dynamic Uneven Margins

Experiments

Conclusions

Setup

- ACE 2007 English corpus:
  - 599 training files, 354 files for EMD testing, 155 for RMD testing
  - 7 entity types subdivided in 44 subtypes
  - 6 relation types with 18 subtypes
- For EMD we detect the entity type, subtype, and entity mention type. The entity class set always to SPC.
- For RMD we detect the relation type, subtype, and direction. Relation modality set always to `Asserted`. Relation tense set to `Unspecified`. We do not detect temporal arguments.
- Parameter tuning:
  - Performed on training using 5-fold cross validation
  - EMD – RMD beam: top 2 classes $+$ beam 100
  - PADUM: $\Gamma_{+1} = 1.0$ and $\Gamma_{-1} = 0.01$
  - Combination parameters: $\lambda_e = 1.0$ and $\lambda_r = 0.5$
  - Combination beam: top 20 candidates from EMD and RMD

EMD Scores for Entity Types

	Count				Cost (%)						
	Ent	Detection		Rec	Detection		Rec	Value	Value-based		
	Tot	FA	Miss	Err	FA	Miss	Err	(%)	Pre	Rec	F
FAC	719	67	244	212	8.6	25.9	14.4	**51.1**	72.2	59.7	**65.3**
GPE	3198	165	385	775	3.6	10.1	10.8	**75.6**	84.7	79.2	**81.8**
LOC	422	50	135	152	10.2	22.9	17.3	**49.6**	68.5	59.8	**63.8**
ORG	2677	157	475	1119	5.8	16.4	14.1	**63.6**	77.7	69.5	**73.4**
PER	10359	560	804	2285	6.9	8.2	1.7	**83.2**	91.3	90.1	**90.7**
VEH	413	16	118	95	3.2	25.7	4.7	**66.4**	89.8	69.6	**78.4**
WEA	335	21	124	136	10.6	42.0	2.6	**44.8**	80.8	55.4	**65.7**
total	18123	1036	2285	4774	5.8	11.8	7.4	**75.0**	85.9	80.8	**83.3**

- Overall: ACE score of 75.0, value-based F score of 83.3 → encouraging results considering the simplicity of the approach; plus we had no coreference resolution and no ACE-specific gazetteers.
- Quantitative analysis: 1 hour/epoch to train. Labels 50 words/second. Without the second-order feature map: 159 seconds/epoch to train and labels 14,000 words/second. Performance drop without the second-order features between 1–3 F1 points.

RMD Scores for Relation Types

	Count				Cost (%)						
	Ent	Detection		Rec	Detection		Rec	Value	Value-based		
	Tot	FA	Miss	Err	FA	Miss	Err	(%)	Pre	Rec	F
ART	261	38	157	84	9.1	63.9	2.5	**24.5**	74.2	33.6	**46.2**
GEN-AFF	235	28	120	92	9.1	51.5	5.0	**34.5**	75.6	43.6	**55.3**
ORG-AFF	503	71	216	237	9.6	45.4	4.0	**41.0**	78.9	50.6	**61.6**
PART-WHOLE	354	57	182	110	12.1	48.9	2.2	**36.8**	77.4	48.9	**59.9**
PER-SOC	213	24	90	116	5.6	38.5	2.4	**53.5**	88.0	59.1	**70.7**
PHYS	428	76	298	113	8.7	69.1	6.2	**16.0**	62.3	24.7	**35.4**
total	1994	294	1063	752	9.4	53.5	4.0	**33.1**	76.1	42.5	**54.5**

- Overall: ACE score of 33.1, value-based F score of 54.5 → ranked as the second organization in the evaluation, within 0.8 ACE points from the best system.
- Quantitative analysis: 47 seconds/epoch to train. Labels 23,000 words/second (assuming labeled entity mentions).

RMD Scores for Relation Subtypes

	Count				Cost (%)						
	Ent	Detection		Rec	Detection		Rec	Value	Value-based		
	Tot	FA	Miss	Err	FA	Miss	Err	(%)	Pre	Rec	F
Artifact	14	0	13	1	0.0	92.0	2.4	**5.6**	70.0	5.6	**10.4**
Business	63	4	39	24	2.2	63.8	3.4	**30.7**	85.6	32.8	**47.5**
Citizen...	171	23	83	73	10.5	49.6	5.7	**34.1**	73.3	44.6	**55.5**
Employment	344	61	113	189	12.1	34.8	4.0	**49.1**	79.1	61.2	**69.0**
Family	118	19	32	79	8.6	20.9	0.4	**70.1**	89.7	78.7	**83.8**
Founder	6	0	5	1	0.0	88.8	3.4	**7.8**	70.0	7.8	**14.1**
Geographical	223	33	102	71	10.4	42.0	1.9	**45.7**	82.1	56.1	**66.7**
Investor...	8	0	5	3	0.0	57.1	2.9	**40.0**	93.3	40.0	**56.0**
Lasting-Personal	32	1	19	13	1.9	50.6	7.8	**39.8**	81.2	41.6	**55.0**
Located	382	72	263	102	9.2	68.3	6.6	**15.9**	61.4	25.1	**35.6**
Membership	96	8	55	33	6.0	61.3	4.2	**28.5**	77.2	34.5	**47.7**
Near	46	4	35	11	4.9	75.2	3.2	**16.7**	72.8	21.6	**33.3**
Org-Location	64	5	37	19	5.9	55.6	3.2	**35.3**	82.0	41.2	**54.8**
Ownership	15	2	13	2	5.0	87.5	0.0	**7.5**	71.4	12.5	**21.3**
Sports-Affiliation	17	0	15	2	0.0	88.4	3.5	**8.1**	70.0	8.1	**14.6**
Student-Alum	17	0	10	7	0.0	60.0	7.5	**32.5**	81.2	32.5	**46.4**
Subsidiary	117	24	67	38	16.1	58.8	2.9	**22.2**	66.8	38.3	**48.7**
User-Owner...	261	38	157	84	9.1	63.9	2.5	**24.5**	74.2	33.6	**46.2**
total	1994	294	1063	752	9.4	53.5	4.0	**33.1**	76.1	42.5	**54.5**

Comparison with Other Systems

System	EMD Score
IBM	82.9
BBN3	81.2
BBN2	81.2
BBN1	81.2
LCC.1	80.9
UPC1	**75.0**
LockheedMartin	67.3
LCC.0	64.4
Fudan	42.3
SAIC_10	12.2
SAIC_8	1.1
SAIC_9	0.2

System	RMD Score
BBN3	33.9
BBN1	33.6
BBN2	33.4
UPC1	**33.1**
LCC1	32.5
LCC0	32.5

Comparison of PADUM vs PA vs SVM

- Experiment: RMD using gold entity mentions on the training corpus with 5-fold cross validation.
- Compared with the standard averaged PA ($\Gamma_{\pm 1} = 0$), and with SVM (C-SVC SVM type, $C = 1.0$; $gamma = 1/k$, where $k = 18$ is the number of categories).

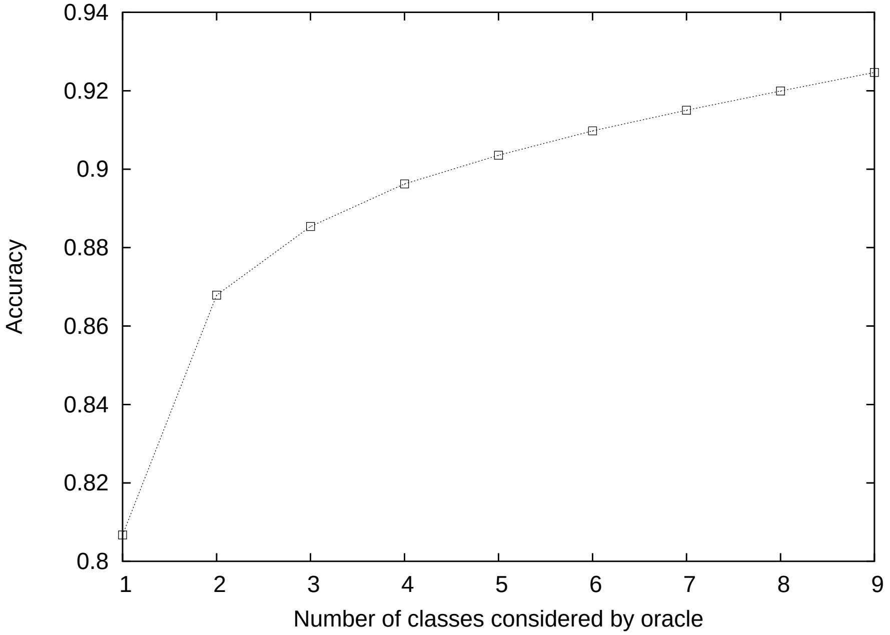
	Precision	Recall	F_1
PADUM, 1 epoch	65.71%	45.48%	53.75
PADUM, 5 epochs	62.96%	56.31%	**59.44**
Avg PA, 1 epoch	**67.94%**	40.28%	50.58
Avg PA, 5 epochs	66.64%	52.19%	58.53
SVM	50.62%	**63.72%**	56.42

- PADUM has better F1 score than both SVM and PA.
- PADUM the most P/R balanced of the three algorithms (without any significant tuning).
- Learning speed: PADUM – needs less than 5 minutes to converge, SVM – 15 hours under the same conditions.

Motivation for the Chosen Architecture: Analysis of Entity Classification

	P	R	F_1
Rec.	92.39%	87.60%	89.93
Rec. + Cls.	77.81%	74.41%	76.07

EMD analysis on the training corpus



Accuracy of the classification oracle

- The major failure point for EMD is entity classification.
- Classification accuracy significantly improved when considering the top two or three classes.

Comparison with Other Architectures

Compared with two typical architectures:

- Pipeline: only the top class output by EMD and RMD.
- (Pseudo) Roth and Yih: inference using Constraint Satisfaction (simulated using our framework). No communication between EMD and RMD.

	Count				Cost (%)						
	Ent	Detection		Rec	Detection		Rec	Value	Value-based		
EMD	Tot	FA	Miss	Err	FA	Miss	Err	(%)	Pre	Rec	F
This paper	54824	2907	5805	16394	5.2	9.0	6.7	**79.1**	87.6	84.3	85.9
Pipeline	54824	2907	5805	16406	5.2	9.0	6.7	**79.0**	87.6	84.2	85.9
Roth & Yih	54824	2907	5805	16400	5.2	9.0	6.7	**79.1**	87.6	84.3	85.9

	Count				Cost (%)						
	Ent	Detection		Rec	Detection		Rec	Value	Value-based		
RMD	Tot	FA	Miss	Err	FA	Miss	Err	(%)	Pre	Rec	F
This paper	8738	1661	4289	3681	12.1	48.7	4.4	**34.8**	74.0	46.9	57.4
Pipeline	8738	1933	4077	3868	14.0	46.6	4.8	**34.6**	72.1	48.6	58.1
Roth & Yih	8738	1310	4865	3244	9.3	55.9	3.7	**31.1**	75.6	40.4	52.7

- Communication between EMD and RMD important: R&Y score 3.7 ACE points lower on RMD.
- Our approach minimally better than Pipeline. But we guarantee a solution consistent with the domain constraints.

Conclusions

- Main focus was simplicity and robustness: all tasks modeled using ML with variants of the PA. We use only syntactic information that can be robustly extracted from text (POS tags and chunks).
- Several contributions:
  - Defined a new Perceptron Algorithm with Dynamic Uneven Margins. Features: large-margin, tailored for class-unbalanced data, adjusts its margins in relation to the generalization performance of the model → performed better than SVM and PA for RMD, even though its training time $\ll$ SVM's.
  - Proposed a strategy to handle errors made in early system stages → when ambiguities detected we let several hypotheses flow though the system and solve them at the end using approximated inference.
- Participated in the 2007 English ACE evaluation for EMD and RMD. Obtained competitive results on both tasks, which is very encouraging considering the simplicity of the approach.