Recent Advances in Chinese Lexical Processing

Weiwei Sun

Institute of Computer Science and Technology
Peking University

July, 2012

Goal – automatically analyzing lexical structures

A raw sentence

警察正在详细调查事故原因

Goal – automatically analyzing lexical structures

A raw sentence

警察正在详细调查事故原因

Word segmentation

警察 / 正在 / 详细 / 调查 / 事故 / 原因

Goal – automatically analyzing lexical structures

A raw sentence

警察正在详细调查事故原因

Word segmentation

警察 / 正在 / 详细 / 调查 / 事故 / 原因

Part-of-speech (POS) tagging

警察/NN 正在/AD 详细/AD 调查/VV 事故/NN 原因/NN

Importance

- Initial steps for Chinese language processing.
- Providing very important information for advanced tasks and applications:
  - Syntactic parsing
  - Semantic parsing
  - Information extraction
  - Information retrieval
  - ...

Outline

Outline

Structured prediction

- The input/output data have a structured and relational form.
- Word segmentation:
  - input: character sequence
  - output: word sequence
- POS tagging:
  - input: word sequence
  - output: POS tag sequence
- Paradigms: kernel methods, structured linear models, graphical models, constrained conditional models, and re-ranking, etc.

Outline

Probabilistic models for sequence pairs

- We have two sequences of random variables: $X_1, X_2, ..., X_n$ and $S_1, S_2, ..., S_n$
- Assume that each S_i is in $\mathcal{S} = \{1, 2, ..., k\}$, and each X_i is in $\mathcal{X} = \{1, 2, ..., o\}$.
- Intuitively,
  - each X_i corresponds to an observation and
  - each S_i corresponds to an underlying state that generated the observation.
- How do we model the joint distribution

$$P(X_1 = x_1, ..., X_n = x_n, S_1 = s_1, ..., S_n = s_n)$$

A history-based model

$$p(x_1...x_n, s_1...s_n; \theta) = p(s_1; \theta) \prod_{j=2}^{n} p(s_j | s_1, ..., s_{j-1}; \theta)$$

- Generate each word from left to right, conditioned on what came before it.
- Very rich representational power
- Too many parameters!

Hidden Markov models

- A HMM takes the following form:

$$p(x_1 \dots x_n, s_1 \dots s_n; \theta) = p(s_1; \theta) \prod_{j=2}^{n} p(s_j | s_{j-1}; \theta) \prod_{j=1}^{n} p(x_j | s_j; \theta)$$

- Parameters in the model:
  1. Initial state parameters ϕ_s for $s \in \mathcal{S}$
  2. Transition parameters $\phi_{s'|s}$ for $s, s' \in \mathcal{S}$
  3. Emission parameters $\eta_{x|s}$ for $s \in \mathcal{S}$ and $x \in \mathcal{X}$
- If we use a specific symbol to denote *stop of a sequence*: $s_0 = *$
  - Denote initial state parameters $\phi_{s|*}$

$$p(x_1 \dots x_n, S_0 = *, s_1 \dots s_n; \theta) = \prod_{j=1}^{n} \left(\phi_{s_j | s_{j-1}} \eta_{x_j | s_j} \right)$$

Higher order HMMs

- We can condition on a longer history of past states:

$$p(*, x_1 ... x_n, s_1 ... s_n; \theta) = \prod_{j=1}^{n} \left(\phi_{(s_j | s_{j-1} ... s_{j-m})} \eta_{x_j | s_j} \right)$$

- A variant:

$$p(*, x_1 ... x_n, s_1 ... s_n; \theta) = \prod_{j=1}^{n} \left(\phi_{(s_j | s_{j-1} ... s_{j-m})} \eta_{(x_j | s_j ... s_{j-l})} \right)$$

- State-of-the-art HMM-based English POS tagging

$$p(*, x_1 ... x_n, s_1 ... s_n; \theta) = \prod_{j=1}^{n} \left(\phi_{(s_j | s_{j-1}, s_{j-2})} \eta_{(x_j | s_j, s_{j-1})} \right)$$

Global linear model

Global Linear Model

$$\hat{\mathbf{y}} = \arg \max_{\mathbf{y} \in \mathrm{GEN}(\mathbf{x})} \theta^\top \Phi(\mathbf{x}, \mathbf{y}) \qquad (1)$$

- Word Segmentation: $\mathbf{x}$ is a sequence of characters, $\mathrm{GEN}(\mathbf{x})$ is the set of possible word sequences.
- Structures are represented via feature mapping $\Phi(\mathbf{x}, \mathbf{y})$
- Parameters θ provide a weight for each feature.
- Direct maximization is generally intractable.

Factored Global Linear Model

$$\hat{\mathbf{y}} = \arg \max_{\mathbf{y} \in \mathrm{GEN}(\mathbf{x})} \sum_{p \in \mathbf{y}} \theta^\top \phi(\mathbf{x}, p) \qquad (2)$$

Global linear models

GLMs for sequences

We score a possible sequence $\mathbf{s}$ for a given sequence $\mathbf{x}$ using linear models:

$$\mathrm{score}(\mathbf{s}, \mathbf{x}) = \mathbf{w}^\top \mathbf{f}(\mathbf{s}, \mathbf{x})$$

To make the a sequence problem solvable, we assume,

$$\mathbf{w}^\top \mathbf{f}(\mathbf{s}, \mathbf{x}) = \sum_{j=1}^{n} \mathbf{w}^\top \mathbf{f}(s_j, s_{j-1}, \mathbf{x}, j)$$

For prediction, we solve the following combinatorial optimization problem:

$$\hat{\mathbf{s}} = \arg\max_{\mathbf{s}} \mathbf{w}^\top \mathbf{f}(\mathbf{s}, \mathbf{x})$$

A probabilistic interpretation

- We then build a giant log-linear model,

$$p(\mathbf{s}|\mathbf{x};\mathbf{w}) = \frac{\exp(\mathbf{w}^\top \mathbf{f}(\mathbf{s},\mathbf{x}))}{\sum_{\mathbf{s}\in\mathcal{S}^n} \exp(\mathbf{w}^\top \mathbf{f}(\mathbf{s},\mathbf{x}))}$$

- The model is giant in the sense that:
  - The space of possible values for s, i.e., $\mathcal{S}^n$, is huge.
  - The normalization constant involves a sum over a huge number of possibilities.
- This model is usually called Conditional Random Fields.

Outline

How to calculate ...

Given the HMM and a sequence:

- The most probable state sequence?
- The probability of the word sequence?
- The (posterior) probability distribution over states, for each word?

Given states and observation sequences, or just observation sequences:

- The parameters of the HMM (ϕ and η)?

Most likely state sequence

- We use an HMM to define

$$p(x_1 \dots x_n, s_1 \dots s_n)$$

  for any sentence $x_1, \dots, x_n$ and state sequence $s_1, \dots, s_n$ of the same length.
- Then the most likely state sequence for $\mathbf{x}$ is

$$\arg\max_{s_1,\dots,s_n} p(x_1 \dots x_n, s_1 \dots s_n)$$

- Statistic view: maximum a *posterior* (MAP) inference
- Computational view: discrete, *combinatorial* optimization

The Viterbi algorithm

- Goal: for a given input sequence $x_1, ..., x_n$, find

$$\arg\max_{s_1,\ldots,s_n} p(x_1 ... x_n, s_1 ... s_n; \theta)$$

- Define

$$r(s_1, ..., s_t) = \prod_{j=1}^{t} \left(\phi_{s_j|s_{j-1}} \eta_{x_j|s_j} \right)$$

- Define a dynamic programming table

  $\pi_t(s)$ = maximum probability of a tag sequence ending in tag s at position t

  that is,

$$\pi_t(s) = \max_{s_1,\ldots,s_{t-1}} r(s_1, ..., s_{t-1}, s)$$

A recursive definition

Recursive definition

- Base case:

$$\pi_1(s) = \phi_{s|*}$$

- For any $t \in \{2, ..., n\}$, for any $s \in \mathcal{S}$:

$$\pi_t(s) = \max_{s' \in \mathcal{S}} (\pi_{t-1}(s') \times \phi_{s|s'} \times \eta_{x_t|s})$$

The Viterbi algorithm

- **Input:** a sentence $x_1, ...x_n$, parameters $\phi_{s'|s}$ and $\eta_{x|s}$
- **Initialization**: Set $\pi_1(s) = \phi_{s|*}$
- **Algorithm**:
  - For $t = 1, ..., n$, for $s \in \mathcal{S}$,

$$\begin{aligned} \pi_t(s) &= \max_{s' \in \mathcal{S}} (\pi_{t-1}(s') \times \phi_{s|s'} \times \eta_{x_t|s}) \\ bp_t(s) &= \arg\max_{s' \in \mathcal{S}} (\pi_{t-1}(s') \times \phi_{s|s'} \times \eta_{x_t|s}) \end{aligned}$$

  - Set

$$s_n = \arg\max_{s' \in \mathcal{S}} \pi_n(s')$$

  - For $t = (n-1), ..., 1$,

$$s_t = bp_{t+1}(s_{t+1})$$

- **Return**:

$$s_1, ..., s_n$$

Probability of the observation sequence

- Goal: for a given input sequence $x_1, ..., x_n$, find

$$p(x_1...x_n; \theta) = \sum_{s_1,...,s_n} p(x_1...x_n, s_1...s_n; \theta)$$

- Define a dynamic programming table

$$\alpha_t(s) \quad = \quad \text{sum of probabilities of all tag sequences ending in tag } s \text{ at position } t$$

that is,

$$\alpha_t(s) = \sum_{s_1,...,s_{t-1}} \left(\prod_{j=1}^{t-1} \phi_{s_j|s_{j-1}} \eta_{x_j|s_j} \right) \times \phi_{s_t|s_{t-1}} \times \eta_{x_t|s_t}$$

The forward algorithm

- **Input:** a sentence $x_1, ...x_n$, parameters $\phi_{s'|s}$ and $\eta_{x|s}$
- **Initialization**: Set $\alpha_1(s) = \phi_{s|*}\eta_{x_1|s}$
- **Algorithm**:
  - For $t = 2, ..., n$,

$$\alpha_t(s) \quad = \quad \sum_{s' \in \mathcal{S}} (\alpha_{t-1}(s') \times \phi_{s|s'} \times \eta_{x_t|s})$$

- **Return**:

$$\sum_{s \in \mathcal{S}} \alpha_n(s)$$

Generalization: Semirings

Only some mathematical properties about operations are relevant $\Rightarrow$ Thinking abstract algebra

- Viterbi and Forward algorithms correspond to exactly the same calculation, except one maximizes and the other sums.
- The same abstract algorithm instantiated in two different semirings.

Semirings

A semiring is a set $\mathcal{A}$ equipped with two binary operations $\oplus$ and $\otimes$, such that:

- $\oplus$ is associative and commutative
  - $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
  - $a \oplus b = b \oplus a$
- $\otimes$ is associative and distributes over $\oplus$
  - $(a \otimes b) \otimes c = a \otimes (b \otimes c)$
  - $a \otimes (b \oplus c) = a \otimes b \oplus a \otimes c$
  - $(a \oplus b) \otimes c = a \otimes c \oplus b \otimes c$
- Identity elements
  - $a \oplus \mathbf{0} = a$
  - $a \otimes \mathbf{1} = a$
  - $a \otimes \mathbf{0} = \mathbf{0} \otimes a = \mathbf{0}$

	Inside	Viterbi
$\mathcal{A}$	$\mathbb{R}_{\geq 0}$	$\mathbb{R}_{\geq 0}$
$\oplus$	$a + b$	$\max(a, b)$
$\otimes$	$a \times b$	$a \times b$
$\mathbf{0}$	0	0
$\mathbf{1}$	1	1

Generalization: Semirings

Generalization

Goal:

$$\bigoplus_{\mathbf{y}\in\mathcal{Y}^n}\left(\bigotimes_{t=2}^{n}\phi(y_t, y_{t-1})\right)$$

A dynamic programming solution:

$$\gamma_t(y) \quad = \quad \bigoplus_{y'\in\mathcal{Y}}\left(\gamma_{t-1}(y')\otimes\phi(Y_t = y, Y_{t-1} = y')\right)$$

Generalization: Semirings

- If

$$\gamma_t(y) = \bigoplus_{y_1 \ldots y_{t-1} \in \mathcal{Y}, Y_t = y} \left(\bigotimes_{j=2}^{t} \phi(y_j, y_{j-1}) \right)$$

- then,

$$\begin{aligned}
\gamma_{t+1}(y) &= \bigoplus_{y' \in \mathcal{Y}} \left(\gamma_t(y') \otimes \phi(Y_{t+1} = y, Y_t = y')\right) \\
&= \bigoplus_{y' \in \mathcal{Y}} \left\{ \left[\bigoplus_{y_1 \ldots y_{t-1} \in \mathcal{Y}, Y_t = y'} \left(\bigotimes_{j=2}^{t} \phi(y_j, y_{j-1}) \right) \right] \otimes \phi(y, y') \right\} \\
&= \bigoplus_{y' \in \mathcal{Y}} \left\{ \bigoplus_{y_1 \ldots y_{t-1} \in \mathcal{Y}, Y_t = y'} \left[\left(\bigotimes_{j=2}^{t} \phi(y_j, y_{j-1}) \right) \otimes \phi(y, y') \right] \right\}
\end{aligned}$$

Generalization: Semirings

$$
\begin{aligned}
\gamma_{t+1}(y) &= \bigoplus_{y' \in \mathcal{Y}} \left\{ \bigoplus_{y_1 \ldots y_{t-1} \in \mathcal{Y}, Y_t = y'} \left[\left(\bigotimes_{j=2}^{t} \phi(y_j, y_{j-1}) \right) \otimes \phi(y, y') \right] \right\} \\
&= \bigoplus_{y' \in \mathcal{Y}} \left(\bigoplus_{y_1 \ldots y_{t-1} \in \mathcal{Y}, Y_t = y', Y_{t+1} = y} \left(\bigotimes_{j=2}^{t+1} \phi(y_j, y_{j-1}) \right) \right) \\
&= \bigoplus_{y_1 \ldots y_t \in \mathcal{Y}, Y_{t+1} = y} \left(\bigotimes_{j=2}^{t+1} \phi(y_j, y_{j-1}) \right)
\end{aligned}
$$

By induction, we can prove,

$$
\bigoplus_{\mathbf{y} \in \mathcal{Y}^n} \left(\bigotimes_{t=2}^{n} \phi(y_t, y_{t-1}) \right) = \bigoplus_{y \in \mathcal{Y}} \gamma_n(y)
$$

The Viterbi algorithm for linear-chain GLMs

Comparison to HMMs

- $\phi_{s'|s}(i) \rightarrow \mathbf{w}^\top \mathbf{f}(s', s)$
- $\eta_{x|s}(i) \rightarrow \mathbf{w}^\top \mathbf{f}(s, \mathbf{x}, i)$

Again, we can use the Viterbi algorithm for decoding.

$$
\begin{aligned}
\arg\max_{\mathbf{s}\in\mathcal{S}^n} p(\mathbf{s}|\mathbf{x};\mathbf{w}) &= \arg\max_{\mathbf{s}\in\mathcal{S}^n} \frac{\exp(\mathbf{w}^\top \mathbf{f}(\mathbf{s},\mathbf{x}))}{\sum_{\mathbf{s}\in\mathcal{S}^n} \exp(\mathbf{w}^\top \mathbf{f}(\mathbf{s},\mathbf{x}))} \\
&= \arg\max_{\mathbf{s}\in\mathcal{S}^n} \exp(\mathbf{w}^\top \mathbf{f}(\mathbf{s},\mathbf{x})) \\
&= \arg\max_{\mathbf{s}\in\mathcal{S}^n} \mathbf{w}^\top \mathbf{f}(\mathbf{s},\mathbf{x}) \\
&= \arg\max_{\mathbf{s}\in\mathcal{S}^n} \sum_{j=1}^{n} \mathbf{w}^\top \mathbf{f}(s_j, s_{j-1}, \mathbf{x}, j)
\end{aligned}
$$

The Viterbi algorithm for linear-chain GLMs

- **Input:** a sentence $x_1, ...x_n$, parameters $\mathbf{w}$.
- **Initialization**: Set $\pi_1(s) = \mathbf{w}^\top \mathbf{f}(s_1, S_0 = *, \mathbf{x}, 1)$
- **Algorithm**:
  - For $t = 1, ..., n$,
    - For $s \in \mathcal{S}$,

$$\begin{aligned} \pi_t(s) &= \max_{s' \in \mathcal{S}} (\pi_{t-1}(s') + \mathbf{w}^\top \mathbf{f}(s, s', \mathbf{x}, t)) \\ bp_t(s) &= \arg\max_{s' \in \mathcal{S}} (\pi_{t-1}(s') + \mathbf{w}^\top \mathbf{f}(s, s', \mathbf{x}, t)) \end{aligned}$$

  - Set

$$s_n = \arg\max_{s' \in \mathcal{S}} \pi_n(s')$$

  - For $t = (n-1), ..., 1$,

$$s_t = bp_{t+1}(s_{t+1})$$

- **Return** the tag sequence $s_1, ..., s_n$.

Outline

Parameter estimation

- Familiar distinction:
  - local learning
  - structured perceptron
  - conditional random fields
  - max-margin Markov network
  - etc.
- Batch learning or online learning
  - structured perceptron
  - margin Infused Related Algorithm (MIRA)
  - SGD optimatization method:
    - stochastic gradient descent
    - sub-gradient descent

Online learning

- A learning algorithm is given a sequence of examples $(x^{(1)}, y^{(1)})$, $(x^{(2)}, y^{(2)})$, ..., $(x^{(m)}, y^{(m)})$ in order.
- The algorithm first sees $x^{(i)}$ and is asked to predict what it thinks $y^{(i)}$ is.
- After making its prediction, the true value of $y^{(i)}$ is revealed, and the algorithm learn something.

In the online learning setting

We are interested in the #error made by the algorithm in total.

Perceptron for binary classification

- Binary classification: $y \in \{-1, +1\}$
- Hypothesis:

$$h_\theta(x) = \text{sign}(\theta^\top x)$$

An iterative learning procedure

For $it = 1, ..., T$, for $i = 1, ..., m$:

- If prediction is wrong, then update θ:

$$\theta^{(k+1)} := \theta^{(k)} + y^{(i)} x^{(i)}$$

- First glance:

$$\underbrace{y^{(i)}(\theta^{(k+1)})^\top x^{(i)}}_{increase} = \underbrace{y^{(i)}(\theta^{(k)})^\top x^{(i)}}_{<0} + \underbrace{||y^{(i)} x^{(i)}||^2}_{\geq 0}$$

Correctness

Theorem

Let a sequence of examples $(x^{(1)}, y^{(1)})$, $(x^{(2)}, y^{(2)})$, ..., $(x^{(m)}, y^{(m)})$ *be given. Suppose that* $||x^{(i)}|| \leq D$ *for all* i*, and further that there exists a unit-length* u *(*$||u||^2 = 1$*) such that* $y^{(i)} \cdot (u^\top x^{(i)}) \geq \gamma$ *for all examples. Then the total number of mistakes that the perceptron algorithm makes on this sequence is at most* $(D/\gamma)^2$*.*

Correctness

What is a good hypothesis θ?

$$\cos(\alpha) = \frac{\theta^\top u}{||\theta|| \cdot ||u||} = \frac{\theta^\top u}{||\theta||} \to 1$$

Both $\theta^\top u$ and $||\theta||$ increase, but $\theta^\top u$ increases faster.

$$\begin{aligned} (\theta^{(k+1)})^\top u &= (\theta^{(k)})^\top u + y^{(i)}(x^{(i)})^\top u \\ &\geq (\theta^{(k)})^\top u + \gamma \end{aligned}$$

We then have,

$$(\theta^{(k)})^\top u \geq k\gamma \tag{3}$$

Correctness

$$
\begin{aligned}
||\theta^{(k+1)}||^2 &= ||\theta^{(k)} + y^{(i)}x^{(i)}||^2 \\
&= ||\theta^{(k)}||^2 + ||x^{(i)}||^2 + 2y^{(i)}(\theta^{(k)})^\top x^{(i)} \\
&\leq ||\theta^{(k)}||^2 + ||x^{(i)}||^2 \\
&\leq ||\theta^{(k)}||^2 + D^2
\end{aligned}
$$

We then have,

$$||\theta^{(k)}||^2 \leq kD^2 \tag{4}$$

Putting Eq. 3 and 4 together and we find that,

$$1 \geq \cos(\alpha) = \frac{(\theta^{(k)})^\top u}{||\theta^{(k)}|| \cdot ||u||} \geq \frac{k\gamma}{\sqrt{k}D}$$

Finally,

$$k \leq (D/\gamma)^2 \tag{5}$$

Structured perceptron for linear-chain GLMs

- Initialization: $\mathbf{w} = \mathbf{0}$.

An iterative learning procedure

For $it = 1, ..., T$, for $i = 1, ..., m$:

- Use the Viterbi algorithm to calculate

$$\hat{\mathbf{s}}^{(i)} = \arg\max_{\mathbf{s}\in\mathcal{S}^n} \mathbf{w}^\top \mathbf{f}(\mathbf{s}, \mathbf{x}^{(i)}) = \arg\max_{\mathbf{s}\in\mathcal{S}^n} \sum_{j=1}^{n} \mathbf{w}^\top \mathbf{f}(s_j, s_{j-1}, \mathbf{x}^{(i)}, j)$$

- Updates:

$$\mathbf{w}^{(k+1)} := \mathbf{w}^{(k)} + \mathbf{f}(\mathbf{s}^{(i)}, \mathbf{x}^{(i)}) - \mathbf{f}(\hat{\mathbf{s}}^{(i)}, \mathbf{x}^{(i)})$$

- Return: $\mathbf{w}$

General structured perceptron

- Initialization: $\mathbf{w} = \mathbf{0}$.

An iterative learning procedure

For $it = 1, ..., T$, for $i = 1, ..., m$:

- Use the Viterbi algorithm to calculate

$$\hat{\mathbf{s}}^{(i)} = \arg \max_{\mathbf{s} \in \mathrm{GEN}(\mathbf{x})} \mathbf{w}^{\top} \mathbf{f}(\mathbf{s}, \mathbf{x}^{(i)})$$

- Updates:

$$\mathbf{w}^{(k+1)} := \mathbf{w}^{(k)} + \mathbf{f}(\mathbf{s}^{(i)}, \mathbf{x}^{(i)}) - \mathbf{f}(\hat{\mathbf{s}}^{(i)}, \mathbf{x}^{(i)})$$

- Return: $\mathbf{w}$

Correctness

Theorem

Let a sequence of examples $(\mathbf{x}^{(1)}, \mathbf{y}^{(1)})$, $(\mathbf{x}^{(2)}, \mathbf{y}^{(2)})$, ..., $(\mathbf{x}^{(m)}, \mathbf{y}^{(m)})$ *be given. Suppose that* $||\mathbf{f}(\mathbf{x}^{(i)}, \mathbf{y})|| \leq D$ *for all* i*, and further that there exists a unit-length* $\mathbf{u}$ *(*$||\mathbf{u}||^2 = 1$*) such that* $\forall \mathbf{y}(\mathbf{u}^\top \mathbf{f}(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) - \mathbf{u}^\top \mathbf{f}(\mathbf{x}^{(i)}, \mathbf{y})) \geq \gamma$ *for all examples. Then the total number of mistakes that the perceptron algorithm makes on this sequence is at most* $(D/\gamma)^2$*.*

$$\begin{aligned}
(\mathbf{w}^{(k+1)})^\top \mathbf{u} &= (\mathbf{w}^{(k)})^\top \mathbf{u} + (\mathbf{f}(\mathbf{x}^{(i)}, \mathbf{s}^{(i)}) - \mathbf{f}(\mathbf{x}^{(i)}, \hat{\mathbf{s}}^{(i)}))^\top \mathbf{u} \\
&\geq (\mathbf{w}^{(k)})^\top \mathbf{u} + \gamma
\end{aligned}$$

We then have,

$$(\mathbf{w}^{(k)})^\top \mathbf{u} \geq k\gamma \tag{6}$$

Correctness

$$\begin{aligned}
||\mathbf{w}^{(k+1)}||^2 &= ||\mathbf{w}^{(k)} + \mathbf{f}(\mathbf{s}^{(i)}, \mathbf{x}^{(i)}) - \mathbf{f}(\hat{\mathbf{s}}^{(i)}, \mathbf{x}^{(i)})||^2 \\
&= ||\mathbf{w}^{(k)}||^2 + ||\mathbf{f}(\mathbf{s}^{(i)}, \mathbf{x}^{(i)}) - \mathbf{f}(\hat{\mathbf{s}}^{(i)}, \mathbf{x}^{(i)})||^2 + \\
&\quad \underbrace{2(\mathbf{w}^{(k)})^\top(\mathbf{f}(\mathbf{s}^{(i)}, \mathbf{x}^{(i)}) - \mathbf{f}(\hat{\mathbf{s}}^{(i)}, \mathbf{x}^{(i)}))}_{wrong\ prediction:\ <0} \\
&< ||\mathbf{w}^{(k)}||^2 + ||\mathbf{f}(\mathbf{s}^{(i)}, \mathbf{x}^{(i)}) - \mathbf{f}(\hat{\mathbf{s}}^{(i)}, \mathbf{x}^{(i)})||^2 \\
&\leq ||\mathbf{w}^{(k)}||^2 + D^2
\end{aligned}$$

We then have,

$$||\mathbf{w}^{(k)}||^2 \leq kD^2$$

Finally,

$$1 \geq \cos(\alpha) = \frac{(\mathbf{w}^{(k)})^\top u}{||\mathbf{w}^{(k)}|| \cdot ||u||} \geq \frac{k\gamma}{\sqrt{k}D} \Rightarrow k \leq (D/\gamma)^2$$

Structured perceptron with averaging

- Initialization: $\mathbf{w} = \mathbf{0}$, $\mathbf{w}_a = \mathbf{0}$.

An iterative learning procedure

For $it = 1, ..., T$, for $i = 1, ..., m$:

- Use the Viterbi algorithm to calculate

$$\hat{\mathbf{s}}^{(i)} = \arg\max_{\mathbf{s} \in \mathcal{S}^n} \mathbf{w}^\top \mathbf{f}(\mathbf{s}, \mathbf{x}^{(i)}) = \arg\max_{\mathbf{s} \in \mathcal{S}^n} \sum_{j=1}^{n} \mathbf{w}^\top \mathbf{f}(s_j, s_{j-1}, \mathbf{x}^{(i)}, j)$$

- Updates:

$$\mathbf{w}^{(k+1)} := \mathbf{w}^{(k)} + \mathbf{f}(\mathbf{s}^{(i)}, \mathbf{x}^{(i)}) - \mathbf{f}(\hat{\mathbf{s}}^{(i)}, \mathbf{x}^{(i)})$$

$$\mathbf{w}_a := \mathbf{w}_a + \mathbf{w}$$

- Return: $\mathbf{w}_a / mT$

Online passive-aggressive algorithms

An iterative learning procedure

For $it = 1, ..., T$, for $i = 1, ..., m$:

- Use the Viterbi algorithm to calculate

$$\hat{\mathbf{s}}^{(i)} = \arg\max_{\mathbf{s}\in\mathcal{S}^n} \mathbf{w}^\top \mathbf{f}(\mathbf{s}, \mathbf{x}^{(i)}) = \arg\max_{\mathbf{s}\in\mathcal{S}^n} \sum_{j=1}^{n} \mathbf{w}^\top \mathbf{f}(s_j, s_{j-1}, \mathbf{x}^{(i)}, j)$$

- Calculate learning rate τ

$$\tau = \frac{\mathrm{loss}(\mathbf{s}^{(i)}, \hat{\mathbf{s}}^{(i)})}{||\mathbf{f}(\mathbf{s}^{(i)}, \mathbf{x}^{(i)}) - \mathbf{f}(\hat{\mathbf{s}}^{(i)}, \mathbf{x}^{(i)})||^2}$$

- Updates:

$$\mathbf{w}^{(k+1)} := \mathbf{w}^{(k)} + \tau(\mathbf{f}(\mathbf{s}^{(i)}, \mathbf{x}^{(i)}) - \mathbf{f}(\hat{\mathbf{s}}^{(i)}, \mathbf{x}^{(i)}))$$

Three variants of the passive-aggressive algorithm

Different learning rates:

$$\begin{aligned}
\tau^{\mathrm{PA}} &= \frac{\mathrm{loss}(\mathbf{s}^{(i)}, \hat{\mathbf{s}}^{(i)})}{||\mathbf{f}(\mathbf{s}^{(i)}, \mathbf{x}^{(i)}) - \mathbf{f}(\hat{\mathbf{s}}^{(i)}, \mathbf{x}^{(i)})||^2} \\
\tau^{\mathrm{PA-I}} &= \min\left\{C, \frac{\mathrm{loss}(\mathbf{s}^{(i)}, \hat{\mathbf{s}}^{(i)})}{||\mathbf{f}(\mathbf{s}^{(i)}, \mathbf{x}^{(i)}) - \mathbf{f}(\hat{\mathbf{s}}^{(i)}, \mathbf{x}^{(i)})||^2}\right\} \\
\tau^{\mathrm{PA-II}} &= \frac{\mathrm{loss}(\mathbf{s}^{(i)}, \hat{\mathbf{s}}^{(i)})}{||\mathbf{f}(\mathbf{s}^{(i)}, \mathbf{x}^{(i)}) - \mathbf{f}(\hat{\mathbf{s}}^{(i)}, \mathbf{x}^{(i)})||^2 + \frac{1}{2C}}
\end{aligned}$$

Pros and cons of online learning

Two types of learning

- Batch learning
- Online learning

Pros:

- Simple to understand
- Easy to implement
- Suitable for large-scale structured learning, which is a often case in NLP.

Cons:

- Theoretically, not as good as batch learning

Outline

Goal – automatically analyzing word boundaries

A raw sentence

警察正在详细调查事故原因

Word segmentation

警察 / 正在 / 详细 / 调查 / 事故 / 原因

Outline

Character-based and Word-based Views

The character-based view

- Basic predicting units: characters
- Character-by-Character

The word-based view

- Basic predicting units: words
- Word-by-word

Key problems:

- How to decide whether a local sequence of characters is a word?
- How to do disambiguation if ambiguous segmentation occurs?

Statistical solutions based on *discriminative structured learning* are popular for both views.

Discriminative Character-based Segmentation

Positional character labels

B	Current character is the start of a multi-character word.
E	Current character is the end of a multi-character word.
I	Current character is a middle of a multi-character word.
S	Current character is a single-character word.

Example

赵	紫	阳	总	理	的	秘	密	日	记
B	I	E	B	E	S	B	E	B	E

Discriminative Character-based Segmentation

The Character-based Model

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}\in\mathcal{Y}^{|\mathbf{c}|}} \theta^\top \Psi(\mathbf{c}, \mathbf{y}) = \arg\max_{\mathbf{y}\in\mathcal{Y}^{|\mathbf{c}|}} \theta^\top \sum_{i=1}^{|\mathbf{c}|} \psi(\mathbf{c}, y_{[i-o:i]}) \tag{7}$$

- It can be seen as a Markov model
- Markov assumption is necessary to follow for computation.
- First-order model is used in our experiments:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}\in\mathcal{Y}^{|\mathbf{c}|}} \theta^\top \sum_{i=1}^{|\mathbf{c}|} \psi(\mathbf{c}, y_{i-1}, y_i)$$

Discriminative Word-based Segmentation

The Word-based Model

$$\hat{\mathbf{w}} = \arg \max_{\mathbf{w} \in \mathrm{GEN}(\mathbf{c})} \theta^{\top} \Phi(\mathbf{c}, \mathbf{w}) = \arg \max_{\mathbf{w} \in \mathrm{GEN}(\mathbf{c})} \theta^{\top} \sum_{i=1}^{|\mathbf{w}|} \phi(\mathbf{c}, w_{[i-o:i]}) \quad (8)$$

- It can be seen as a semi-Markov model
- Markov assumption is necessary to follow for computation.
- First-order model is used in our experiments:

$$\hat{\mathbf{w}} = \arg \max_{\mathbf{w} \in \mathrm{GEN}(\mathbf{c})} \theta^{\top} \sum_{i=1}^{|\mathbf{w}|} \phi(\mathbf{c}, w_{i-1}, w_i)$$

Outline

Internal Structure of Words

- The character structure, i.e. word formation, of a word is important.
- E.g., character "者/person" is usually used as a suffix meaning "one kind of people".
- Word formation information is not well explored in word-based models.
- Character-based models partially characterize the internal structures. Discriminative latent variable CRF works.

Linearity and Nonlinearity

- In general, a sequence classification itself involves nonlinearity: The features of current token usually encode previous state information.
- This kind of nonlinearity exists in both word-based and character-based models.
- In the character-based model, the inductive way for word prediction behaves nonlinearly: The character label sequence for a word is either "BI*E" or "S".

Dynamic Tokens or Static Tokens

- In the word-based model, the processing units, i.e. predicted words, are not fixed.
- In the character-based model, the processing units, i.e. characters, are not fixed.
- The upper bound of the score $\sum_{i=1}^{|\mathbf{w}|}$ increases while more words are separated.
- Word-based segmenter tends to segment words into smaller pieces.

Word Token or Word Type Features

- Two kinds of "words"
  - Words in dictionary are word types;
  - Words in sentences are word tokens.
- The character-based model
  - Features are usually defined by the character n-grams.
  - It is slightly less natural to encode predicted word token information.
  - Character-based segmenters can use word type features by looking up a dictionary.
- The word-based model
  - Taking words as dynamic tokens, it is very easy to define word token features in a word-based model.
  - Word-based segmenters hence have greater representational power.

Upper Bound of System Combination

- The error analysis suggests that there is still space for improvement, just by combining the two existing models.
- Upper bound: Let the two segmenters vote with the oracle segmenter.

	P(%)	R(%)	F	ER (%)
AS	96.6	96.9	96.7	37.7
CU	97.4	97.1	97.3	46.0
MSR	97.5	97.7	97.6	35.1
PKU	96.8	96.2	96.5	32.7

Table: Upper bound for combination. The error reduction (ER) rate is a comparison between the F-score produced by the oracle combination system and the character-based system.

Method

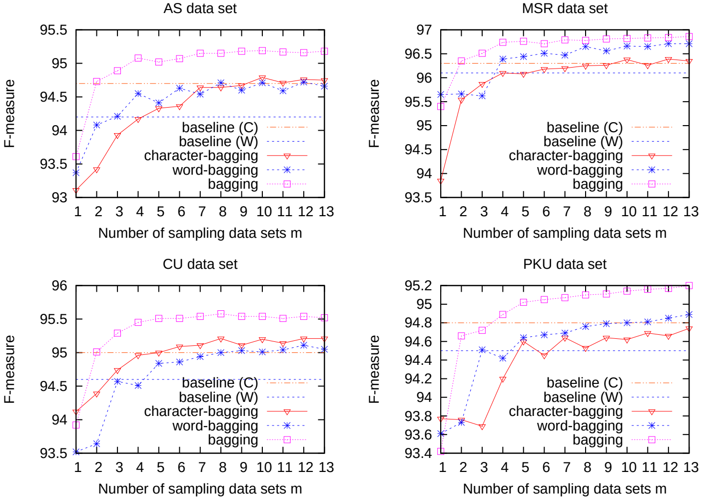
Bootstrap aggregating: a machine learning ensemble meta-algorithm.

- Given a training set D of size n, Bagging generates m new training sets D_i of size $n' \leq n$, by sampling examples from D uniformly.
- The m models are fitted using the above m bootstrap samples and combined by voting.

A Bagging model to combine segmentaters

- Generates m new training sets D_i of size $63.2\% \times n$ by sampling.
- Each D_i is separately used to train a word-based segmenter and a character-based segmenter.
- In the segmentation phase, $2m$ models outputs $2m$ segmentation results.
- The final segmentation is the voting result.

Results


AS data set
MSR data set
CU data set
PKU data set
F-measure
Number of sampling data sets m
1 2 3 4 5 6 7 8 9 10 11 12 13
95.5
95
94.5
94
93.5
93
97
96.5
96
95.5
95
94.5
94
93.5
96
95.5
95
94.5
94
93.5
95.2
95
94.8
94.6
94.4
94.2
94
93.8
93.6
93.4
baseline (C)
baseline (W)
character-bagging
word-bagging
bagging

Outline

Motivations

- Supervised NLP: Blah blah blah ...
- Unsupervised NLP: Blah blah blah ...
- Soooooo labeled data + unlabeled data: Blah blah blah ...
- *Note that* we focus on improving strong supervised segmenters trained on large-scale labeled data.

Unlabeled Data

Three types of unlabeled data:

- Large-scale in-domain data ♠
- Large-scale out-of-domain data [domain adaptation]
- Small-scale target document ♠

Related machine learning topics

- Semi-supervised learning
- Transductive learning

Transductive, Document-level Word Segmentation

- Many applications involve processing a whole document.
- The text of the current document can provide additional useful information to segment a sentence.

Example: “氨纶丝”

- As a translated terminology word, it lacks compositionality.
- As a result, if it does not appear in the training data, it is very hard for statistical models to recognize this word.

Transductive, Document-level Word Segmentation

- Many applications involve processing a whole document.
- The text of the current document can provide additional useful information to segment a sentence.

Example: "氨纶丝"

- As a translated terminology word, it lacks compositionality.
- As a result, if it does not appear in the training data, it is very hard for statistical models to recognize this word.

中国最大的氨伦丝生产基地——钟山氨伦有限公司，日前在连云港开发区建成并投产。 这个采用差别化氨伦丝生产技术改造的项目，总投资七千万元，累计年产氨纶丝一千五百吨。 连云港钟山氨纶丝有限公司是九十年代初引进日本东洋纺织技术和设备的一家苏港合资企业，前两期工程建成并投产以来，已累计实现利润一点四七亿元，跻身于国家二类大型企业行列。 这次新投产的生产线，由该公司自行设计，自行开发，自行调试。 设备从安装到投产只用了三个月时间，开发了企业自己的专利技术，为公司下一步对外输出氨伦丝生产技术奠定了基础。

Feature Induction

- A general framework for semi-supervised NLP.
- Strategy: Derive informative features from unlabeled data and use them in discriminative models.
- Simple? YES!
- Effective? OFTEN!
- The models: Compact and easier to interpret.
- Examples: Named entity recognition, dependency parsing, etc.

Derived Features

A candidate character token c_i with a context ...$c_{i-1}c_ic_{i+1}$....

- Mutual information:

$$MI(c_{[i-2:i-1]}, c_{[i:i+1]}) = \log \frac{p(c_{[i-2:i+1]})}{p(c_{[i-2:i-1]})p(c_{[i:i+1]})}$$

- Accessor variety:
  - *Left accessor variety*: the number of distinct characters that precede s
  - *Right accessor variety*: the number of distinct characters that succeed s
- Punctuation variety:
  - *Left punctuation variety*: the number of times a punctuation precedes s in a corpus
  - *Right punctuation variety*: the number of how many times a punctuation succeeds s

Derived Features

A candidate character token c_i with a context $\ldots c_{i-1}c_ic_{i+1}\ldots$.

- *String count*: the number of times a given string appears in that document. Our document-based features include,
  - Whether the string count of $c_{[s:i]}$ is equal to that of $c_{[s:i+1]}$ $(i-3 \leq s \leq i)$.
  - Whether the string count of $c_{[i:e]}$ is equal to that of $c_{[i-1:e]}$ $(i \leq e \leq i+3)$.

Intuitions

- Accessor variety: When a string appears under different linguistic environments, it may carry a meaning.
- Punctuation as anchor words: Punctuation marks can be taken as perfect word delimiters.
- Document-based features: The string counts of $c_{[s:i]}$ and $c_{[s:i+1]}$ being equal means that when $c_{[s:i]}$ appears, it appears inside $c_{[s:i+1]}$ and that $c_{[s:i]}$ is not independently used in this document.

Main Results

Devel.	$F_{\beta=1}$	R_{oov}
Baseline	95.46	77.68
+MI	95.49	77.98
+AV(2)	95.94	79.31
+AV(2,3)	96.07	80.61
+AV(2,3,4)	96.07	81.83
+PU(2)	95.97	79.70
+PU(2,3)	96.11	80.42
+PU(2,3,4)	96.10	80.53
+MI+AV(2,3,4)+PU(2,3,4)	96.19	80.42
+DOC	95.66	79.89
+MI+AV(2,3,4)+PU(2,3,4)+DOC	96.22	81.75

Main Results

Devel.	$F_{\beta=1}$	R_{oov}
Baseline	95.46	77.68
+MI	95.49	77.98
+AV(2)	95.94	79.31
+AV(2,3)	96.07	80.61
+AV(2,3,4)	96.07	81.83
+PU(2)	95.97	79.70
+PU(2,3)	96.11	80.42
+PU(2,3,4)	96.10	80.53
+MI+AV(2,3,4)+PU(2,3,4)	96.19	80.42
+DOC	95.66	79.89
+MI+AV(2,3,4)+PU(2,3,4)+DOC	96.22	81.75

Main Results

Devel.	$F_{\beta=1}$	R_{oov}
Baseline	95.46	77.68
+MI	95.49	77.98
+AV(2)	95.94	79.31
+AV(2,3)	96.07	80.61
+AV(2,3,4)	96.07	81.83
+PU(2)	95.97	79.70
+PU(2,3)	96.11	80.42
+PU(2,3,4)	96.10	80.53
+MI+AV(2,3,4)+PU(2,3,4)	96.19	80.42
+DOC	95.66	79.89
+MI+AV(2,3,4)+PU(2,3,4)+DOC	96.22	81.75

Main Results

Devel.	$F_{\beta=1}$	R_{oov}
Baseline	95.46	77.68
+MI	95.49	77.98
+AV(2)	95.94	79.31
+AV(2,3)	96.07	80.61
+AV(2,3,4)	96.07	81.83
+PU(2)	95.97	79.70
+PU(2,3)	96.11	80.42
+PU(2,3,4)	96.10	80.53
+MI+AV(2,3,4)+PU(2,3,4)	96.19	80.42
+DOC	95.66	79.89
+MI+AV(2,3,4)+PU(2,3,4)+DOC	96.22	81.75

Main Results

Devel.	$F_{\beta=1}$	R_{oov}
Baseline	95.46	77.68
+MI	95.49	77.98
+AV(2)	95.94	79.31
+AV(2,3)	96.07	80.61
+AV(2,3,4)	96.07	81.83
+PU(2)	95.97	79.70
+PU(2,3)	96.11	80.42
+PU(2,3,4)	96.10	80.53
+MI+AV(2,3,4)+PU(2,3,4)	96.19	80.42
+DOC	95.66	79.89
+MI+AV(2,3,4)+PU(2,3,4)+DOC	96.22	81.75

Main Results

Devel.	$F_{\beta=1}$	R_{oov}
Baseline	95.46	77.68
+MI	95.49	77.98
+AV(2)	95.94	79.31
+AV(2,3)	96.07	80.61
+AV(2,3,4)	96.07	81.83
+PU(2)	95.97	79.70
+PU(2,3)	96.11	80.42
+PU(2,3,4)	96.10	80.53
+MI+AV(2,3,4)+PU(2,3,4)	96.19	80.42
+DOC	95.66	79.89
+MI+AV(2,3,4)+PU(2,3,4)+DOC	96.22	81.75

Main Results

Devel.	$F_{\beta=1}$	R_{oov}
Baseline	95.46	77.68
+MI	95.49	77.98
+AV(2)	95.94	79.31
+AV(2,3)	96.07	80.61
+AV(2,3,4)	96.07	81.83
+PU(2)	95.97	79.70
+PU(2,3)	96.11	80.42
+PU(2,3,4)	96.10	80.53
+MI+AV(2,3,4)+PU(2,3,4)	96.19	80.42
+DOC	95.66	79.89
+MI+AV(2,3,4)+PU(2,3,4)+DOC	96.22	81.75

Main Results

Devel.	$F_{\beta=1}$	R_{oov}
Baseline	95.46	77.68
+MI	95.49	77.98
+AV(2)	95.94	79.31
+AV(2,3)	96.07	80.61
+AV(2,3,4)	96.07	81.83
+PU(2)	95.97	79.70
+PU(2,3)	96.11	80.42
+PU(2,3,4)	96.10	80.53
+MI+AV(2,3,4)+PU(2,3,4)	96.19	80.42
+DOC	95.66	79.89
+MI+AV(2,3,4)+PU(2,3,4)+DOC	96.22	81.75

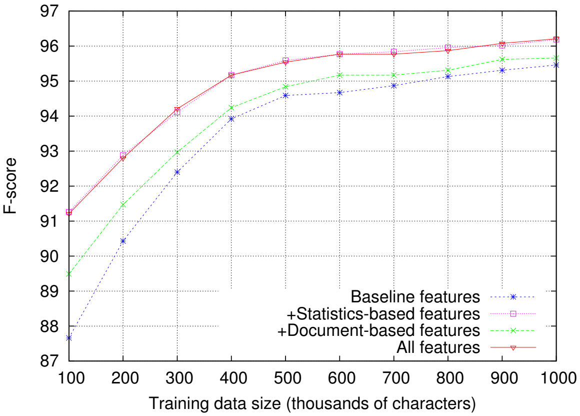
Main Results

Devel.	$F_{\beta=1}$	R_{oov}
Baseline	95.46	77.68
+MI	95.49	77.98
+AV(2)	95.94	79.31
+AV(2,3)	96.07	80.61
+AV(2,3,4)	96.07	81.83
+PU(2)	95.97	79.70
+PU(2,3)	96.11	80.42
+PU(2,3,4)	96.10	80.53
+MI+AV(2,3,4)+PU(2,3,4)	96.19	80.42
+DOC	95.66	79.89
+MI+AV(2,3,4)+PU(2,3,4)+DOC	96.22	81.75

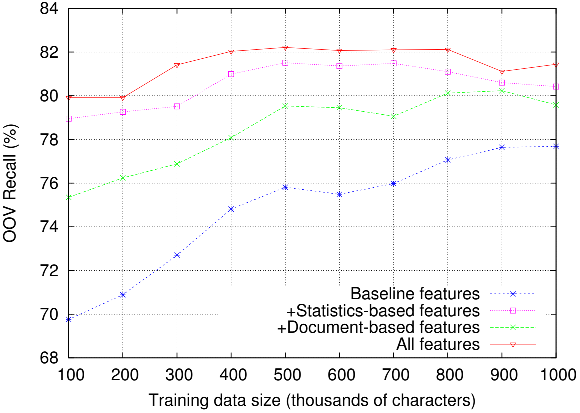
Main Results

Devel.	$F_{\beta=1}$	R_{oov}
Baseline	95.46	77.68
+MI	95.49	77.98
+AV(2)	95.94	79.31
+AV(2,3)	96.07	80.61
+AV(2,3,4)	96.07	81.83
+PU(2)	95.97	79.70
+PU(2,3)	96.11	80.42
+PU(2,3,4)	96.10	80.53
+MI+AV(2,3,4)+PU(2,3,4)	96.19	80.42
+DOC	95.66	79.89
+MI+AV(2,3,4)+PU(2,3,4)+DOC	96.22	81.75

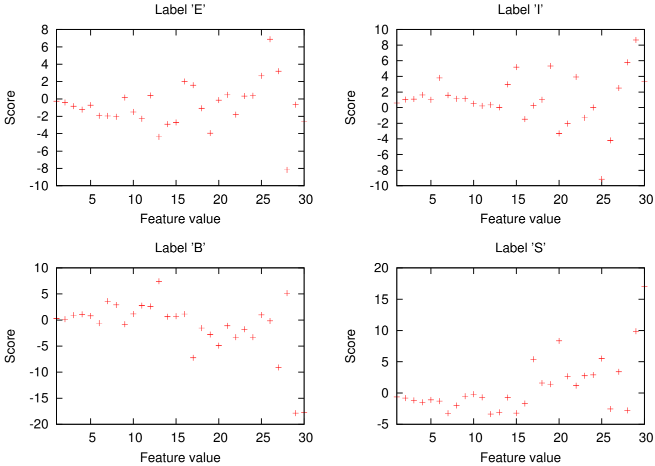
Learning Curves



Learning Curves



Binary (☺) or Numeric (☹) Features



Outline

Goal – automatically analyzing lexical structures

Word segmentation

警察 / 正在 / 详细 / 调查 / 事故 / 原因

Part-of-speech (POS) tagging

警察/NN 正在/AD 详细/AD 调查/VV 事故/NN 原因/NN

Outline

State-of-the-art methods

Discriminative sequence labeling based methods achieve the state-of-the-art of English POS tagging. (ACL wiki)

Averaged Perceptron	Averaged Perception discriminative sequence model	Collins (2002)
Maxent easiest-first	Maximum entropy bidirectional easiest-first inference	Tsuruoka and Tsujii (2005)
SVMTool	SVM-based tagger and tagger generator	Giménez and Márquez (2004)
Morče/COMPOST	Averaged Perceptron	Spoustová et al. (2009)
Stanford Tagger 1.0	Maximum entropy cyclic dependency network	Toutanova et al. (2003)
Stanford Tagger 2.0	Maximum entropy cyclic dependency network	Manning (2011)
Stanford Tagger 2.0	Maximum entropy cyclic dependency network	Manning (2011)
LTAG-spinal	Bidirectional perceptron learning	Shen et al. (2007)

State-of-the-art methods

- Structured prediction techniques, especially global linear models.
  - Structured perceptron
  - Conditional random fields
- It is easy to utilize rich features
  - Word form features
  - Morphological features
- It is easy to explore other information sources by designing new features.
  - Extra dictionaries

A state-of-the-art system

Features

- Word uni-grams
- Word bi-grams
- Prefix strings
- Suffix strings

A state-of-the-art system

Features for w_i in $\ldots w_{i-2} w_{i-1} w_i w_{i+1} w_{i+2} \ldots$:

- Word uni-grams
- Word bi-grams
- Prefix strings
- Suffix strings

A state-of-the-art system

Features for w_i in $\ldots w_{i-2} w_{i-1} w_i w_{i+1} w_{i+2} \ldots$:

- Word uni-grams: w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}
- Word bi-grams
- Prefix strings
- Suffix strings

A state-of-the-art system

Features for w_i in $\ldots w_{i-2} w_{i-1} w_i w_{i+1} w_{i+2} \ldots$:

- Word uni-grams: w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}
- Word bi-grams: $w_{i-2}w_{i-1}$, $w_{i-1}w_i$, w_iw_{i+1}, $w_{i+1}w_{i+2}$
- Prefix strings
- Suffix strings

A state-of-the-art system

Features for $w_i = c_1 \ldots c_n$ in $\ldots w_{i-2} w_{i-1} w_i w_{i+1} w_{i+2} \ldots$:

- Word uni-grams: w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}
- Word bi-grams: $w_{i-2}w_{i-1}$, $w_{i-1}w_i$, w_iw_{i+1}, $w_{i+1}w_{i+2}$
- Prefix strings
- Suffix strings

A state-of-the-art system

Features for $w_i = c_1 ... c_n$ in $... w_{i-2} w_{i-1} w_i w_{i+1} w_{i+2} ...$:

- Word uni-grams: w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}
- Word bi-grams: $w_{i-2}w_{i-1}$, $w_{i-1}w_i$, w_iw_{i+1}, $w_{i+1}w_{i+2}$
- Prefix strings: c_1, c_1c_2, $c_1c_2c_3$
- Suffix strings

A state-of-the-art system

Features for $w_i = c_1 ... c_n$ in $... w_{i-2} w_{i-1} w_i w_{i+1} w_{i+2} ...$:

- Word uni-grams: w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}
- Word bi-grams: $w_{i-2}w_{i-1}$, $w_{i-1}w_i$, $w_i w_{i+1}$, $w_{i+1}w_{i+2}$
- Prefix strings: c_1, c_1c_2, $c_1c_2c_3$
- Suffix strings: c_n, $c_{n-1}c_n$, $c_{n-2}c_{n-1}c_n$

A state-of-the-art system

Features for $w_i = c_1 ... c_n$ in $... w_{i-2} w_{i-1} w_i w_{i+1} w_{i+2} ...$:

- Word uni-grams: w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}
- Word bi-grams: $w_{i-2}w_{i-1}$, $w_{i-1}w_i$, w_iw_{i+1}, $w_{i+1}w_{i+2}$
- Prefix strings: c_1, c_1c_2, $c_1c_2c_3$
- Suffix strings: c_n, $c_{n-1}c_n$, $c_{n-2}c_{n-1}c_n$

Discriminative sequential tagging achieves the state-of-the-art of Chinese POS tagging.

System	Acc.
Trigram HMM (Huang et al., 2009)	93.99%
Bigram HMM-LA (Huang et al., 2009)	94.53%
Discriminative sequential tagging	94.69%

A state-of-the-art system

Features for $w_i = c_1 ... c_n$ in $... w_{i-2} w_{i-1} w_i w_{i+1} w_{i+2} ...$:

- Word uni-grams: w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}
- Word bi-grams: $w_{i-2}w_{i-1}$, $w_{i-1}w_i$, w_iw_{i+1}, $w_{i+1}w_{i+2}$
- Prefix strings: c_1, c_1c_2, $c_1c_2c_3$
- Suffix strings: c_n, $c_{n-1}c_n$, $c_{n-2}c_{n-1}c_n$

Discriminative sequential tagging achieves the state-of-the-art of Chinese POS tagging.

System	Acc.
Trigram HMM (Huang et al., 2009)	93.99%
Bigram HMM-LA (Huang et al., 2009)	94.53%
Discriminative sequential tagging	94.69%

A state-of-the-art system

Example

刘华清
副总理
的
这
次
来访

A state-of-the-art system

Example

刘华清
副总理
的
这
次
来访

A state-of-the-art system

Example

刘华清
副总理
的
这
次
来访

A state-of-the-art system

Example

刘华清
副总理
的
这
次
来访

A state-of-the-art system

Example

	Prefix	Suffix
刘华清		
副总理	P1:副;P2:副总;P3:副总理	S1:理;S2:总理;S3:副总理
的		
这		
次		
来访		

A state-of-the-art system

Example

	Prefix	Suffix	POS
刘华清			
副总理	P1:副;P2:副总;P3:副总理	S1:理;S2:总理;S3:副总理	NN
的			
这			
次			
来访			

A state-of-the-art system

Example

	Prefix	Suffix	POS
刘华清	P1:刘;P2:刘华;P3:刘华清	S1:清;S2:华清;S3:刘华清	
副总理	P1:副;P2:副总;P3:副总理	S1:理;S2:总理;S3:副总理	
的	P1:的	S1:的	
这	P1:这	S1:这	
次	P1:次	S1:次	
来访	P1:来;P2:来访	S1:访;S2:来访	

A state-of-the-art system

Example

	Prefix	Suffix	POS
刘华清	P1:刘;P2:刘华;P3:刘华清	S1:清;S2:华清;S3:刘华清	NR
副总理	P1:副;P2:副总;P3:副总理	S1:理;S2:总理;S3:副总理	NN
的	P1:的	S1:的	DEG
这	P1:这	S1:这	DT
次	P1:次	S1:次	M
来访	P1:来;P2:来访	S1:访;S2:来访	NN

Error analysis I

Word frequency	Acc.
0 [Unknown word]	83.55%
1-5	89.31%
6-10	90.20%
11-100	94.88%
101-1000	96.26%
1001-	93.65%

Tagging accuracies relative to word frequency.

Error analysis I

Word frequency	Acc.
0 [Unknown word]	83.55%
1-5	89.31%
6-10	90.20%
11-100	94.88%
101-1000	96.26%
1001-	93.65%

Classifiction of low-frequency words is hard.

Error analysis I

Word frequency	Acc.
0 [Unknown word]	83.55%
1-5	89.31%
6-10	90.20%
11-100	94.88%
101-1000	96.26%
1001-	93.65%

Classifiction of very high-frequency words is hard too.

Error analysis II

- A word projects its grammatical property to its maximal projection.
- A maximal projection syntactically governs all words under it.
- The words under the span of current token thus reflect its syntactic behavior and are good clues for POS tagging.

Length of span	Acc.
1-2	93.79%
3-4	93.39%
5-6	92.19%

Tagging accuracies relative to span length.

Error analysis II

- A word projects its grammatical property to its maximal projection.
- A maximal projection syntactically governs all words under it.
- The words under the span of current token thus reflect its syntactic behavior and are good clues for POS tagging.

Length of span	Acc.	
1-2	93.79%	
3-4	93.39%	$\downarrow$
5-6	92.19%	$\downarrow$

$\#\{$words governed by a word$\}\uparrow$;

Error analysis II

- A word projects its grammatical property to its maximal projection.
- A maximal projection syntactically governs all words under it.
- The words under the span of current token thus reflect its syntactic behavior and are good clues for POS tagging.

Length of span	Acc.	
1-2	93.79%	
3-4	93.39%	$\downarrow$
5-6	92.19%	$\downarrow$

$\#\{$words governed by a word$\}\uparrow$; the prediction difficulty $\uparrow$

What a linguist say?

- Meaning arises from the differences between linguistic units.
- These differences are of two kinds:
    - paradigmatic: concerning substitution
    - syntagmatic: concerning positioning
- Functions:
    - paradigmatic: differentiation
    - syntagmatic: possibilities of combination
- The distinction is a key one in structuralist semiotic analysis.

What a linguist say?

- The *value* of a word is determined by both paradigmatic and syntagmatic lexical relations.
- Both relations have a great impact on POS tagging.

What a linguist say?

- The *value* of a word is determined by both paradigmatic and syntagmatic lexical relations.
- Both relations have a great impact on POS tagging.

Low tagging accuracy of low-frequency words

Lack of knowledge about paradigmatic lexical relations.

What a linguist say?

- The *value* of a word is determined by both paradigmatic and syntagmatic lexical relations.
- Both relations have a great impact on POS tagging.

Low tagging accuracy of low-frequency words

Lack of knowledge about paradigmatic lexical relations.

Low tagging accuracy of words governing long spans

Lack of information about syntagmatic lexical relations.

Outline

Word clustering

Word clustering

Partitioning sets of words into subsets of syntactically or semantically similar words.

- A very useful technique to capture paradigmatic or substitutional similarity among words.
  - Unsuperivsed word clustering explores paradigmatic lexical relations encoded in unlabeled data.
    - A great quantity of unlabeled data can be used $\Rightarrow$ We can automatically acquire a large lexicon
- To bridge the gap between high and low frequency words, word clusters are utilized as features.

Clustering algorithms

Distributional word clustering

Words that appear in similar contexts tend to have similar meanings.

Based on the word bi-gram context:

- Brown clustering

$$P(w_i|w_1, ...w_{i-1}) \approx p(C(w_i)|C(w_{i-1}))p(w_i|C(w_i))$$

- MKCLS clustering

$$P(w_i|w_1, ...w_{i-1}) \approx p(C(w_i)|w_{i-1})p(w_i|C(w_i))$$

Brown and MKCLS Clustering

- Hard clustering: each word belongs to exactly one cluster.
- Good open source tools.
- Successful application to boost named entity recognition and dependency parsing.

Main results

Features	Brown	MKCLS
Supervised	94.48%	
+ #100	94.82%	94.93%
+ #500	94.92%	94.99%
+ #1000	94.90%	95.00%

Main results

Features	Brown	MKCLS
Supervised	94.48%	
+ #100	94.82%↑	94.93%↑
+ #500	94.92%↑	94.99%↑
+ #1000	94.90%↑	95.00%↑

Consistently improved.

Main results

Features	Brown	MKCLS
Supervised	94.48%	
+ #100	94.82%↑	94.93%↑
+ #500	94.92%↑	94.99%↑
+ #1000	94.90%↑	95.00%↑

Consistently improved.

The granularities do not affect much.

Main results

Features	Brown	MKCLS
Supervised	94.48%	
+ #100	94.82%↑	94.93%↑
+ #500	94.92%↑	94.99%↑
+ #1000	94.90%↑	95.00%↑

Consistently improved.

The granularities do not affect much.

Combine different clustering algorithms

- + Brown features + MKCLS features

Combine different granularities of clusters

- + #100 + #500 + #1000

Main results

Features	Brown	MKCLS
Supervised	94.48%	
+ #100	94.82%↑	94.93%↑
+ #500	94.92%↑	94.99%↑
+ #1000	94.90%↑	95.00%↑

Consistently improved.

The granularities do not affect much.

Combine different clustering algorithms

- + Brown features + MKCLS features ⇒ No further improvement

Combine different granularities of clusters

- + #100 + #500 + #1000 ⇒ No further improvement

Supervised or semi-supervised word segmentation

To cluster Chinese words, we must segment raw texts first.

- Supervised segmenter: a traditional character-based segmenter.
- Semi-supervised segmenter: a character-based segmenter with
  - string knowledges that are automatically induced from unlabeled data.

Features	Segmenter	MKCLS
+ #100	Supervised	94.83%
+ #500	Supervised	94.93%
+ #1000	Supervised	94.95%
+ #100	Semi-supervised	94.97%
+ #500	Semi-supervised	94.88%
+ #1000	Semi-supervised	94.94%

Supervised or semi-supervised word segmentation

To cluster Chinese words, we must segment raw texts first.

- Supervised segmenter: a traditional character-based segmenter.
- Semi-supervised segmenter: a character-based segmenter with
    - string knowledges that are automatically induced from unlabeled data.

Features	Segmenter	MKCLS
+ #100	Supervised	94.83%
+ #500	Supervised	94.93%
+ #1000	Supervised	94.95%
+ #100	Semi-supervised	94.97%
+ #500	Semi-supervised	94.88%
+ #1000	Semi-supervised	94.94%

No significant difference.

Learning curves

Size	Baseline	+Cluster
4.5K	90.10%	91.93%
9K	92.91%	93.94%
13.5K	93.88%	94.60%
18K	94.24%	94.77%
22K	94.48%	95.00%

Learning curves

Size	Baseline	+Cluster
4.5K	90.10%	91.93% ↑
9K	92.91%	93.94% ↑
13.5K	93.88%	94.60% ↑
18K	94.24%	94.77% ↑
22K	94.48%	95.00% ↑

Consistently improved.

Two-fold contribution

- Word clustering abstracts context information.
  - This linguistic knowledge is helpful to better correlate a word in a certain context to its POS tag.
- The clustering of the unknown words fights the sparse data.
  - Correlate an unknown word with known words through their classes.

Supervised	94.48%
+Known words' clusters	94.70%
+All words' clusters	95.02%

Evaluation

Two-fold contribution

- Word clustering abstracts context information.
  - This linguistic knowledge is helpful to better correlate a word in a certain context to its POS tag.
- The clustering of the unknown words fights the sparse data.
  - Correlate an unknown word with known words through their classes.

Supervised	94.48%	
+Known words' clusters	94.70%	$\uparrow$0.22
+All words' clusters	95.02%	

Useful linguistic knowledge.

Two-fold contribution

- Word clustering abstracts context information.
  - This linguistic knowledge is helpful to better correlate a word in a certain context to its POS tag.
- The clustering of the unknown words fights the sparse data.
  - Correlate an unknown word with known words through their classes.

Supervised	94.48%	
+Known words' clusters	94.70%	$\uparrow$0.22
+All words' clusters	95.02%	$\uparrow$0.32

Fight the data sparse problem.

Tagging recall of unknown words

	Baseline	+Clustering	Δ
AD	33.33%	42.86%	
CD	97.99%	98.39%	
JJ	3.49%	26.74%	
NN	91.05%	91.34%	
NR	81.69%	88.76%	
NT	60.00%	68.00%	
VA	33.33%	53.33%	
VV	67.66%	72.39%	

Tagging recall of unknown words

	Baseline	+Clustering	Δ
AD	33.33%	42.86%	$<$
CD	97.99%	98.39%	$<$
JJ	3.49%	26.74%	$<$
NN	91.05%	91.34%	$<$
NR	81.69%	88.76%	$<$
NT	60.00%	68.00%	$<$
VA	33.33%	53.33%	$<$
VV	67.66%	72.39%	$<$

The recall of all *unknown words* is improved.

Outline

Capturing syntagmatic lexical relations

- Syntax-free discriminative sequential tagging:
  - Flexible to integrate multiple informance sources.
    - Like word clustering.
  - Reach state-of-the-art [94.48%]
- Syntax-based generative chart parsing:
  - Rely on treebanks.
  - Close to state-of-the-art [93.69%]
- Syntactic structures $\Rightarrow$ Syntagmatic lexical relations

Complementary strengths

A comparative analysis illuminates more precisely the contribution of full syntactic information in Chinese POS tagging.

☺Tagger$>$ ☹Parser	☹Tagger$<$☺Parser
open classes	close classes
content words	function words
local disambiguation	global disambiguation

Empirical comparison

Parser<Tagger		Parser>Tagger	
AD	94.15<94.71	AS	98.54>98.44
CD	94.66<97.52	BA	96.15>92.52
CS	91.12<92.12	CC	93.80>90.58
ETC	99.65<100.0	DEC	85.78>81.22
JJ	81.35<84.65	DEG	88.94>85.96
LB	91.30<93.18	DER	80.95>77.42
LC	96.29<97.08	DEV	84.89>74.78
M	95.62<96.94	DT	98.28>98.05
NN	93.56<94.95	MSP	91.30>90.14
NR	89.84<95.07	P	96.26>94.56
NT	96.70<97.26	VV	91.99>91.87
OD	81.06<86.36		
PN	98.10<98.15		
SB	95.36<96.77		
SP	61.70<68.89		
VA	81.27<84.25		
VC	95.91<97.67		
VE	97.12<98.48		

Overall

Tagger:	94.48%
Parser:	93.69%

- Open classes vs.close classes

Empirical comparison

Parser<Tagger		Parser>Tagger	
AD	94.15<94.71	AS	98.54>98.44
CD	94.66<97.52	BA	96.15>92.52
CS	91.12<92.12	CC	93.80>90.58
ETC	99.65<100.0	DEC	85.78>81.22
JJ	81.35<84.65	DEG	88.94>85.96
LB	91.30<93.18	DER	80.95>77.42
LC	96.29<97.08	DEV	84.89>74.78
M	95.62<96.94	DT	98.28>98.05
NN	93.56<94.95	MSP	91.30>90.14
NR	89.84<95.07	P	96.26>94.56
NT	96.70<97.26	VV	91.99>91.87
OD	81.06<86.36		
PN	98.10<98.15		
SB	95.36<96.77		
SP	61.70<68.89		
VA	81.27<84.25		
VC	95.91<97.67		
VE	97.12<98.48		

Overall

Tagger:	94.48%
Parser:	93.69%

- Open classes vs.close classes

Empirical comparison

Parser<Tagger		Parser>Tagger	
AD	94.15<94.71	AS	98.54>98.44
CD	94.66<97.52	BA	96.15>92.52
CS	91.12<92.12	CC	93.80>90.58
ETC	99.65<100.0	DEC	85.78>81.22
JJ	81.35<84.65	DEG	88.94>85.96
LB	91.30<93.18	DER	80.95>77.42
LC	96.29<97.08	DEV	84.89>74.78
M	95.62<96.94	DT	98.28>98.05
NN	93.56<94.95	MSP	91.30>90.14
NR	89.84<95.07	P	96.26>94.56
NT	96.70<97.26	VV	91.99>91.87
OD	81.06<86.36		
PN	98.10<98.15		
SB	95.36<96.77		
SP	61.70<68.89		
VA	81.27<84.25		
VC	95.91<97.67		
VE	97.12<98.48		

Overall

Tagger:	94.48%
Parser:	93.69%

- Open classes vs.close classes

	Known	Unknown
Tagger	95.22%	81.59%
Parser	95.38%	64.77%

Empirical comparison

Parser<Tagger		Parser>Tagger	
AD	94.15<94.71	AS	98.54>98.44
CD	94.66<97.52	BA	96.15>92.52
CS	91.12<92.12	CC	93.80>90.58
ETC	99.65<100.0	DEC	85.78>81.22
JJ	81.35<84.65	DEG	88.94>85.96
LB	91.30<93.18	DER	80.95>77.42
LC	96.29<97.08	DEV	84.89>74.78
M	95.62<96.94	DT	98.28>98.05
NN	93.56<94.95	MSP	91.30>90.14
NR	89.84<95.07	P	96.26>94.56
NT	96.70<97.26	VV	91.99>91.87
OD	81.06<86.36		
PN	98.10<98.15		
SB	95.36<96.77		
SP	61.70<68.89		
VA	81.27<84.25		
VC	95.91<97.67		
VE	97.12<98.48		

Overall

Tagger:	94.48%
Parser:	93.69%

- Open classes vs.close classes

	Known	Unknown
Tagger	95.22%	81.59%
Parser	95.38%	64.77%

Empirical comparison

Parser<Tagger		Parser>Tagger	
AD	94.15<94.71	AS	98.54>98.44
CD	94.66<97.52	BA	96.15>92.52
CS	91.12<92.12	CC	93.80>90.58
ETC	99.65<100.0	DEC	85.78>81.22
JJ	81.35<84.65	DEG	88.94>85.96
LB	91.30<93.18	DER	80.95>77.42
LC	96.29<97.08	DEV	84.89>74.78
M	95.62<96.94	DT	98.28>98.05
NN	93.56<94.95	MSP	91.30>90.14
NR	89.84<95.07	P	96.26>94.56
NT	96.70<97.26	VV	91.99>91.87
OD	81.06<86.36		
PN	98.10<98.15		
SB	95.36<96.77		
SP	61.70<68.89		
VA	81.27<84.25		
VC	95.91<97.67		
VE	97.12<98.48		

Overall

Tagger:	94.48%
Parser:	93.69%

- Open classes vs.close classes

	Known	Unknown
Tagger	95.22%	81.59%
Parser	95.38%	64.77%

- Content words vs. function words

Empirical comparison

Parser<Tagger		Parser>Tagger	
AD	94.15<94.71	AS	98.54>98.44
CD	94.66<97.52	BA	96.15>92.52
CS	91.12<92.12	CC	93.80>90.58
ETC	99.65<100.0	DEC	85.78>81.22
JJ	81.35<84.65	DEG	88.94>85.96
LB	91.30<93.18	DER	80.95>77.42
LC	96.29<97.08	DEV	84.89>74.78
M	95.62<96.94	DT	98.28>98.05
NN	93.56<94.95	MSP	91.30>90.14
NR	89.84<95.07	P	96.26>94.56
NT	96.70<97.26	VV	91.99>91.87
OD	81.06<86.36		
PN	98.10<98.15		
SB	95.36<96.77		
SP	61.70<68.89		
VA	81.27<84.25		
VC	95.91<97.67		
VE	97.12<98.48		

Overall

Tagger:	94.48%
Parser:	93.69%

- Open classes vs.close classes

	Known	Unknown
Tagger	95.22%	81.59%
Parser	95.38%	64.77%

- Content words vs. function words
- Local disambiguation vs. global disambiguation

Model ensemble

- Model ensemble: voting?

Model ensemble

- Model ensemble: voting?
- Oops! Only two systems.

Model ensemble

- Model ensemble: voting?
- Oops! Only two systems.
- Let's generate more sub-models.

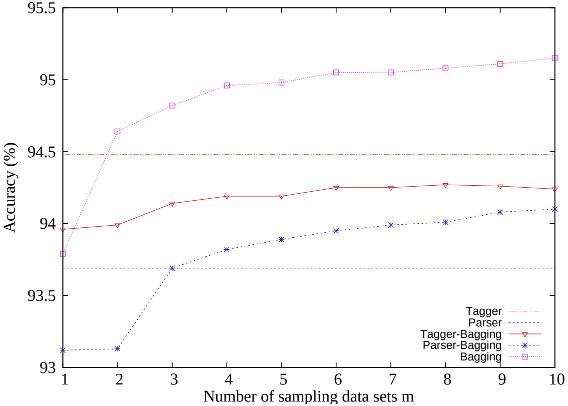
Model ensemble

- Model ensemble: voting?
- Oops! Only two systems.
- Let's generate more sub-models.

A Bagging model

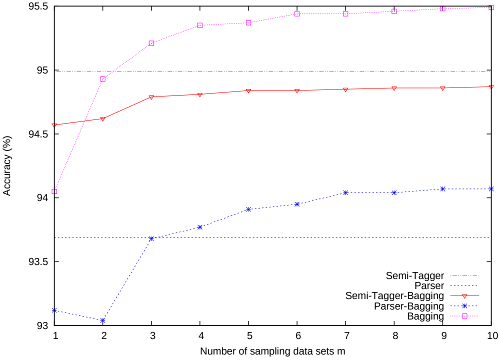
- Generating m new training sets D_i by sampling. [Bootstrap]
- Each D_i is separately used to train a tagger and a parser.
- In the test phase, $2m$ models outputs $2m$ tagging results
- The final prediction is the voting result. [Aggregating]

Results


95.5
95
94.5
94
93.5
93
1
2
3
4
5
6
7
8
9
10
Accuracy (%)
Number of sampling data sets m
Tagger
Parser
Tagger-Bagging
Parser-Bagging
Bagging

Combining both

- Two distinguished improvements: capturing different types of lexical relations
- Further improvement: combining both



Final results

Tagger	94.33%
Tagger+Parser	94.96%
Tagger[+cluster]	94.85%
Tagger[+cluster]+Parser	95.34%

Evaluation

Final results

Tagger	94.33%
Tagger+Parser	94.96%
Tagger[+cluster]	94.85%
Tagger[+cluster]+Parser	95.34%

Baseline achieves state-of-the-art

Final results

Tagger	94.33%
Tagger+Parser	94.96%
Tagger[+cluster]	94.85%
Tagger[+cluster]+Parser	95.34%

Model ensemble helps capture syntagmatic lexical relations

Final results

Tagger	94.33%
Tagger+Parser	94.96%
Tagger[+cluster]	94.85%
Tagger[+cluster]+Parser	95.34%

Learning ensemble helps capture paradigmatic lexical relations

Final results

Tagger	94.33%
Tagger+Parser	94.96%
Tagger[+cluster]	94.85%
Tagger[+cluster]+Parser	95.34%

Two enhancements are not much overlapping

Outline

Goal – automatically analyzing lexical structures

A raw sentence

警察正在详细调查事故原因

Part-of-speech (POS) tagging

警察/NN 正在/AD 详细/AD 调查/VV 事故/NN 原因/NN

The Difficulty and The Motivation

- Joint approaches outperform pipeline approaches in word segmentation and POS tagging.
- A challenge for joint approaches is the large combined search space, which makes efficient decoding very hard.

Our work is motivated by several characteristics of this problem

- A majority of words are easy to identify in the segmentation problem.
- The capability to represent rich contextual features is crucial to a POS tagger.
- Segmenters designed with different views have complementary strength.

A stacked sub-word tagging model

A stacked sub-word tagging model

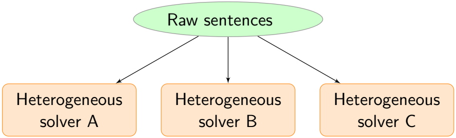
System architecture

A stacked sub-word tagging model

Heterogeneous solves

- trained on heterogeneously labeled data.
- Single view

System architecture

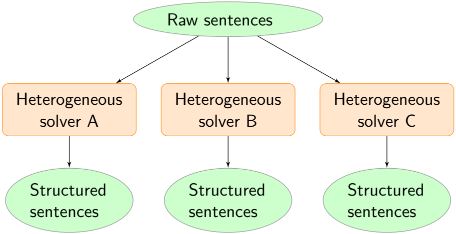


A stacked sub-word tagging model

Structured sentences

- segmented
- tagged

System architecture

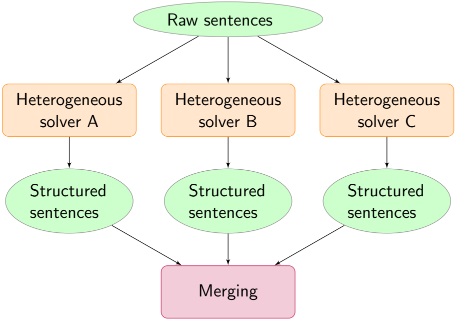


A stacked sub-word tagging model

Merging :

- Maximizing agreements of non-word-breaks
- If two continuous characters are separated by any solver, it is taken as a sub-word break.

System architecture

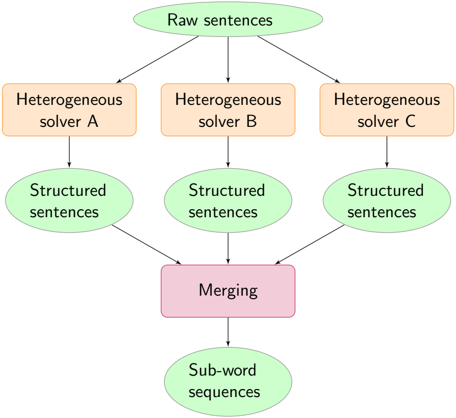


A stacked sub-word tagging model

Sub-words are

- as large as possible
- compatible with all segmentation

System architecture

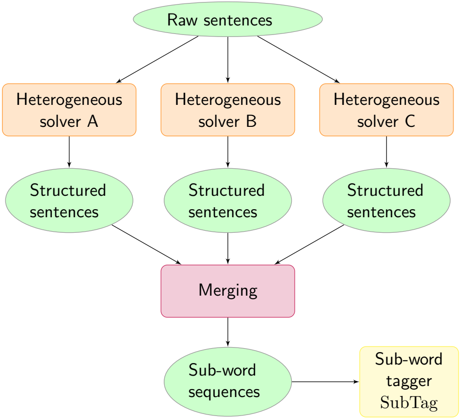


A stacked sub-word tagging model

Good sub-word tagging

- good segmentation
- good POS tagging

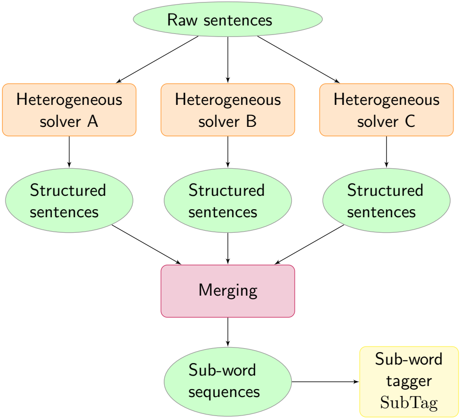
System architecture



A stacked sub-word tagging model

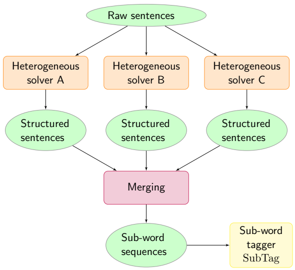
Also a statistical corpus converter.

System architecture



Practical issues

System architecture



Level 0 solvers:

- Same data, different models
- Different data, same model

Process target annotations to generate training data for the level 1 solver:

- heterogeneous level 0 solvers
- homogeneous level 0 solver: cross-validation/stacking

An example

Example

刘华清副总理的这次来访

An example

Example

刘	B-nr
华	B-nr
清	I-nr
副	B-b
总	B-n
理	I-n
的	B-u
这	B-r
次	I-r
来	B-v
访	I-v

An example

Example

刘	B-nr	B-NR
华	B-nr	I-NR
清	I-nr	I-NR
副	B-b	B-NN
总	B-n	I-NN
理	I-n	I-NN
的	B-u	B-DEC
这	B-r	B-DT
次	I-r	B-M
来	B-v	B-NN
访	I-v	I-NN

An example

Example

刘	B-nr	B-NR	刘
华	B-nr	I-NR	华清
清	I-nr	I-NR	
副	B-b	B-NN	副
总	B-n	I-NN	总理
理	I-n	I-NN	
的	B-u	B-DEC	的
这	B-r	B-DT	这
次	I-r	B-M	次
来	B-v	B-NN	来访
访	I-v	I-NN	

An example

Example

刘	B-nr	B-NR	刘	B-nr	B-NR
华	B-nr	I-NR	华清	B-nr	I-NR
清	I-nr	I-NR			
副	B-b	B-NN	副	B-b	B-NN
总	B-n	I-NN	总理	B-n	I-NN
理	I-n	I-NN			
的	B-u	B-DEC	的	B-u	B-DEC
这	B-r	B-DT	这	B-r	B-DT
次	I-r	B-M	次	I-r	B-M
来	B-v	B-NN	来访	B-v	B-NN
访	I-v	I-NN			

An example

Example

刘	B-nr	B-NR	刘	B-nr	B-NR
华	B-nr	I-NR	华清	B-nr	I-NR
清	I-nr	I-NR			
副	B-b	B-NN	副	B-b	B-NN
总	B-n	I-NN	总理	B-n	I-NN
理	I-n	I-NN			
的	B-u	B-DEC	的	B-u	B-DEC
这	B-r	B-DT	这	B-r	B-DT
次	I-r	B-M	次	I-r	B-M
来	B-v	B-NN	来访	B-v	B-NN
访	I-v	I-NN			

An example

Example

刘	B-nr	B-NR	刘	B-nr	B-NR
华	B-nr	I-NR	华清	B-nr	I-NR
清	I-nr	I-NR			
副	B-b	B-NN	副	B-b	B-NN
总	B-n	I-NN	总理	B-n	I-NN
理	I-n	I-NN			
的	B-u	B-DEC	的	B-u	B-DEC
这	B-r	B-DT	这	B-r	B-DT
次	I-r	B-M	次	I-r	B-M
来	B-v	B-NN	来访	B-v	B-NN
访	I-v	I-NN			

An example

Example

刘	B-nr	B-NR	刘	B-nr	B-NR
华	B-nr	I-NR	华清	B-nr	I-NR
清	I-nr	I-NR			
副	B-b	B-NN	副	B-b	B-NN
总	B-n	I-NN	总理	B-n	I-NN
理	I-n	I-NN			
的	B-u	B-DEC	的	B-u	B-DEC
这	B-r	B-DT	这	B-r	B-DT
次	I-r	B-M	次	I-r	B-M
来	B-v	B-NN	来访	B-v	B-NN
访	I-v	I-NN			

An example

Example

刘	B-nr	B-NR	刘	B-nr	B-NR	
华	B-nr	I-NR	华清	B-nr	I-NR	
清	I-nr	I-NR				
副	B-b	B-NN	副	B-b	B-NN	
总	B-n	I-NN	总理	B-n	I-NN	I-NN
理	I-n	I-NN				
的	B-u	B-DEC	的	B-u	B-DEC	
这	B-r	B-DT	这	B-r	B-DT	
次	I-r	B-M	次	I-r	B-M	
来	B-v	B-NN	来访	B-v	B-NN	
访	I-v	I-NN				

An example

Example

刘	B-nr	B-NR	刘	B-nr	B-NR	B-NR
华	B-nr	I-NR	华清	B-nr	I-NR	I-NR
清	I-nr	I-NR				
副	B-b	B-NN	副	B-b	B-NN	B-NN
总	B-n	I-NN	总理	B-n	I-NN	I-NN
理	I-n	I-NN				
的	B-u	B-DEC	的	B-u	B-DEC	B-DEG
这	B-r	B-DT	这	B-r	B-DT	B-DT
次	I-r	B-M	次	I-r	B-M	B-M
来	B-v	B-NN	来访	B-v	B-NN	B-NN
访	I-v	I-NN				

An example

Example

刘	B-nr	B-NR	刘	B-nr	B-NR	B-NR	刘华清/NR
华	B-nr	I-NR	华清	B-nr	I-NR	I-NR	
清	I-nr	I-NR					
副	B-b	B-NN	副	B-b	B-NN	B-NN	副总理/NN
总	B-n	I-NN	总理	B-n	I-NN	I-NN	
理	I-n	I-NN					
的	B-u	B-DEC	的	B-u	B-DEC	B-DEG	的/DEG
这	B-r	B-DT	这	B-r	B-DT	B-DT	这/DT
次	I-r	B-M	次	I-r	B-M	B-M	次/M
来	B-v	B-NN	来访	B-v	B-NN	B-NN	来访/NN
访	I-v	I-NN					

About training

$\boxed{\mathcal{D}_B}$ $\Rightarrow$ Train Level 0 B-style tagger T_B^0

About training

$\mathcal{D}_B$	$\Rightarrow$ Train Level 0 B-style tagger T_B^0
$\mathcal{D}_A$	$\Rightarrow$ Train Level 0 A-style tagger T_A^0

About training

$\mathcal{D}_B$	$\Rightarrow$ Train Level 0 B-style tagger T_B^0
$\mathcal{D}_A$	$\Rightarrow$ Train Level 0 A-style tagger T_A^0
$\mathcal{D}_A \Rightarrow \hat{B}(\mathcal{D}_A)$	$\Rightarrow$ Label $\mathcal{D}_A$ with T_B^0

About training

$\mathcal{D}_B$	$\Rightarrow$ Train Level 0 B-style tagger T_B^0
$\mathcal{D}_A$	$\Rightarrow$ Train Level 0 A-style tagger T_A^0
$\mathcal{D}_A \Rightarrow \hat{B}(\mathcal{D}_A)$	$\Rightarrow$ Label $\mathcal{D}_A$ with T_B^0
$\mathcal{D}_A^{(1)}$ $\mathcal{D}_A^{(2)}$ $\mathcal{D}_A^{(3)}$	$\Rightarrow$ Cross-validation

About training

$\boxed{\mathcal{D}_B}$	$\Rightarrow$ Train Level 0 B-style tagger T_B^0
$\boxed{\mathcal{D}_A}$	$\Rightarrow$ Train Level 0 A-style tagger T_A^0
$\boxed{\mathcal{D}_A \Rightarrow \hat{B}(\mathcal{D}_A)}$	$\Rightarrow$ Label $\mathcal{D}_A$ with T_B^0
Test / Train / Train	$\Rightarrow$ Get $\hat{A}(\mathcal{D}_A^{(1)})$

About training

$\mathcal{D}_B$	$\Rightarrow$ Train Level 0 B-style tagger T_B^0
$\mathcal{D}_A$	$\Rightarrow$ Train Level 0 A-style tagger T_A^0
$\mathcal{D}_A \Rightarrow \hat{B}(\mathcal{D}_A)$	$\Rightarrow$ Label $\mathcal{D}_A$ with T_B^0

Test	Train	Train	$\Rightarrow$ Get $\hat{A}(\mathcal{D}_A^{(1)})$
Train	Test	Train	$\Rightarrow$ Get $\hat{A}(\mathcal{D}_A^{(2)})$

About training

$\mathcal{D}_B$	$\Rightarrow$ Train Level 0 B-style tagger T_B^0
$\mathcal{D}_A$	$\Rightarrow$ Train Level 0 A-style tagger T_A^0
$\mathcal{D}_A \Rightarrow \hat{B}(\mathcal{D}_A)$	$\Rightarrow$ Label $\mathcal{D}_A$ with T_B^0

Test	Train	Train	$\Rightarrow$ Get $\hat{A}(\mathcal{D}_A^{(1)})$
Train	Test	Train	$\Rightarrow$ Get $\hat{A}(\mathcal{D}_A^{(2)})$
Train	Train	Test	$\Rightarrow$ Get $\hat{A}(\mathcal{D}_A^{(3)})$

About training

$\mathcal{D}_B$	$\Rightarrow$ Train Level 0 B-style tagger T_B^0
$\mathcal{D}_A$	$\Rightarrow$ Train Level 0 A-style tagger T_A^0
$\mathcal{D}_A \Rightarrow \hat{B}(\mathcal{D}_A)$	$\Rightarrow$ Label $\mathcal{D}_A$ with T_B^0

Test	Train	Train	$\Rightarrow$ Get $\hat{A}(\mathcal{D}_A^{(1)})$
Train	Test	Train	$\Rightarrow$ Get $\hat{A}(\mathcal{D}_A^{(2)})$
Train	Train	Test	$\Rightarrow$ Get $\hat{A}(\mathcal{D}_A^{(3)})$

$\mathcal{D}_A, \hat{A}(\mathcal{D}_A), \hat{B}(\mathcal{D}_A)$	$\Rightarrow$ Train Level 1 A-style sub-word tagger T_A^1

Annotation Ensemble

- Many NLP systems rely on large-scale, manually annotated corpora.
- Linguistic annotations are
  - important to train statistical models
  - very expensive to build
- Multiple heterogeneous annotations EXIST!
  - Parsing: Penn Treebank vs. Redwoods Treebank
  - Semantic role labeling: Propbank vs. FrameNet
- Different projects $\rightarrow$ different linguistic theories $\rightarrow$ different annotation schemes
- How to consume heterogeneous annotations?

Annotation ensemble

How to consume heterogeneous annotations?

Two essential characteristics

Annotation ensemble

How to consume heterogeneous annotations?

Two essential characteristics

Annotation ensemble

How to consume heterogeneous annotations?

Two essential characteristics

1. Heterogeneous annotations are *different*.

Annotation ensemble

How to consume heterogeneous annotations?

Two essential characteristics

1. Heterogeneous annotations are *different*.
    - Different projects, different linguistic theories, different representation formalisms, different annotation schemes, etc.

Annotation ensemble

How to consume heterogeneous annotations?

Two essential characteristics

1. Heterogeneous annotations are *different*.
   - Different projects, different linguistic theories, different representation formalisms, different annotation schemes, etc.
2. Heterogeneous annotations are *similar*.

Annotation ensemble

How to consume heterogeneous annotations?

Two essential characteristics

1. Heterogeneous annotations are *different*.
   - Different projects, different linguistic theories, different representation formalisms, different annotation schemes, etc.
2. Heterogeneous annotations are *similar*.
   - Same high-level linguistic principles.

Annotation ensemble

How to consume heterogeneous annotations?

Two essential characteristics

1. Heterogeneous annotations are (similar but) *different*.
   - Different projects, different linguistic theories, different representation formalisms, different annotation schemes, etc.
2. Heterogeneous annotations are (different but) *similar*.
   - Same high-level linguistic principles.

Annotation ensemble

How to consume heterogeneous annotations?

Two essential characteristics

1. Heterogeneous annotations are (similar but) *different*.
   - Different projects, different linguistic theories, different representation formalisms, different annotation schemes, etc.
2. Heterogeneous annotations are (different but) *similar*.
   - Same high-level linguistic principles.

- The approximation error: intrinsic suboptimality of the model
- The estimation error: having only finite training data

Annotation ensemble

How to consume heterogeneous annotations?

Two essential characteristics

1. Heterogeneous annotations are (similar but) *different*.
   - Different projects, different linguistic theories, different representation formalisms, different annotation schemes, etc.

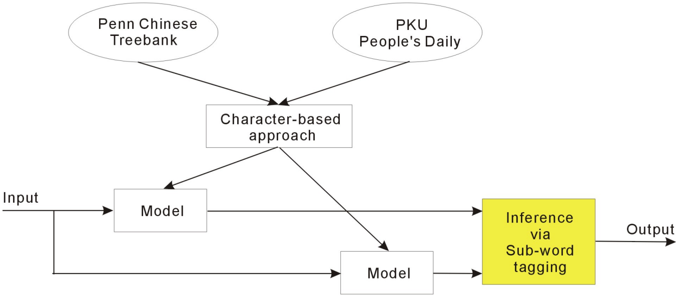
   ☺ Reducing approximation errors

2. Heterogeneous annotations are (different but) *similar*.
   - Same high-level linguistic principles.

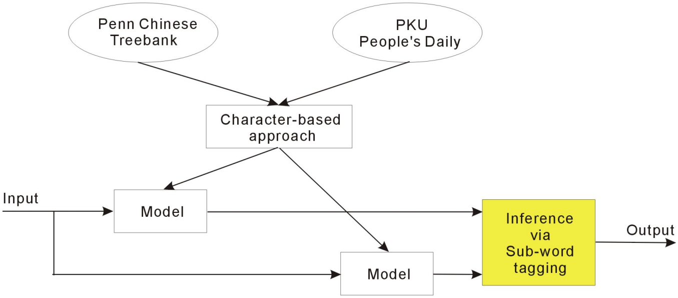
   ☺ Reducing estimation errors

- The approximation error: intrinsic suboptimality of the model
- The estimation error: having only finite training data

Annotation Ensemble



Annotation Ensemble



- Reducing approximation errors
  - Stacking [Feature / structure]
- Reducing estimation errors
  - Corpus conversion [Stacking model is a statistical converter]
  - Model retraining

Main results

We focus on improving CTB-style tagging with PPD.

Test	F-score
State-of-the-art	94.02
Base model	93.41
+Re-training	94.11
Sub-word model	94.36
+Re-training	94.68

F-scores of different systems.

Main results

We focus on improving CTB-style tagging with PPD.

Test	F-score
State-of-the-art	94.02
Base model	93.41
+Re-training	94.11
Sub-word model	94.36
+Re-training	94.68

Stacking model works! Approximation error is reduced!

Main results

We focus on improving CTB-style tagging with PPD.

Test	F-score
State-of-the-art	94.02
Base model	93.41
+Re-training	94.11
Sub-word model	94.36
+Re-training	94.68

Corpus conversion works! Estimation error is reduced!

Main results

We focus on improving CTB-style tagging with PPD.

Test	F-score
State-of-the-art	94.02
Base model	93.41
+Re-training	94.11
Sub-word model	94.36
+Re-training	94.68

Corpus conversion works! Estimation error is reduced!

Main results

We focus on improving CTB-style tagging with PPD.

Test	F-score
State-of-the-art	94.02
Base model	93.41
+Re-training	94.11
Sub-word model	94.36
+Re-training	94.68

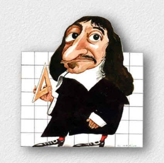
Better than previous results.

Outline

Conclusion

- Recent advances in Chinese lexical processing:
  - Linguistics-inspired improvements
  - Machine learning techniques
- Multi-view processing is important:
  - Heterogeneous models
  - Heterogeneous annotations

Descartes' opinion



The diversity of our opinions does not spring from some of us being more able to reason than others, but only from our conducting our thoughts along different lines and not examining the same things.

Game over

Game over

QUESTIONS?

COMMENTS?