Training Global Linear Models for Chinese Word Segmentation

Dong Song and **Anoop Sarkar**
Natural Language Lab
Simon Fraser University

Introduction

- English: words are separated by space
- Chinese: no space between words
- Word segmentation is important in various natural language processing tasks
  - For example, it is required for Chinese-English machine translation
- Word segmentation is hard:

北京大学生比赛

- 北京(Beijing)/大学生(university students)/比赛(competition)
  Competition among university students in Beijing

- 北京大学(Beijing University)/生(give birth to)/比赛(competition)
  ? Beijing University gives birth to the competition

Global Linear Models for Chinese Word Segmentation

- Find the most plausible word segmentation y' for an un-segmented Chinese sentence x:

$$y' = \underbrace{\arg\max_{y \in GEN(x)}}_{\text{Possible segmentations}} \underbrace{\left(\sum_k \overbrace{w_k}^{\text{Feature weight}} \cdot \overbrace{f_k(x,y)}^{\text{Features of candidate y}}\right)}_{\text{Score for each segmentation}}$$

- Global linear models (Collins, 2002) can be trained using perceptron (voted or averaged variants); max-margin methods; and even CRFs, by normalizing the score above to give log(p(y|x))

Example

$$y' = \arg\max_{y \in GEN(x)} \left(\sum_k w_k \cdot f_k(x, y) \right)$$

- x : 我们生活在信息时代 (we live in an information age)
- GEN(x): y_1, y_2
  - y_1 : 我们(we) / 生活(live) / 在(in) / 信息(information) / 时代(age)
  - y_2 : 我们(we) / 生(born) / 活(alive) / 在(in) / 信息时代(information age)
- w :

f_1	f_2	f_3	f_4	f_5
生活(live)	生(born)	(我们(we), 生活(live))	(我们(we), 生(born))	(信息(information), 时代(age))
$w_1 = 1$	$w_2 = -1$	$w_3 = 2$	$w_4 = -1$	$w_5 = 3$

- For y_1, score = $w_1 f_1 + w_3 f_3 + w_5 f_5$ = 1*1 +2*1 + 3*1 = 6
- For y_2, score = $w_2 f_2 + w_4 f_4$ = -1*1 +(-1)*1= -2
- Thus, $y' = y_1$

Global Linear Models for Chinese Word Segmentation

- In a global linear model, a feature can be global in two ways:
  - It is the sum of local features
    - E.g. feature *word bigram (f_3, f_4, or f_5)* in the entire training corpus
  - It is a holistic feature that cannot be decomposed
    - E.g. *sentence confidence score*
    - To distinguish it with the first meaning, we use the quotation: “global feature”

Global Linear Models for Chinese Word Segmentation

- A global linear model is easy to understand and to implement, but there are many choices in the implementation.
  - E.g. Set of features, training methods
- It is difficult to train weights for “global features”
  - Decomposition
  - Scaling
- We want to find the choices that lead to state of the art accuracy for Chinese Word Segmentation

Contribution of Our Paper

- Compare various methods for learning weights for features that are full sentence features
- Compare re-ranking with full beam search
- Compare an Averaged Perceptron global linear model with a max-margin global linear model (Exponentiated Gradient)

Feature Templates

- Local Feature Template (Zhang and Clark, 2007)

word	1	word w
	2	word bigram w_1w_2
character	3	single character word w
	4	space-separated characters c_1 and c_2
	5	character bi-gram c_1c_2 in any word
character and length	6	a word starting with character c and having length l
	7	a word ending with character c and having length l
word and character	8	the first and last characters c_1 and c_2 of any word
	9	word w immediately before character c
	10	character c immediately before word w
	11	the starting characters c_1 and c_2 of two consecutive words
	12	the ending characters c_1 and c_2 of two consecutive words
word and length	13	a word of length l and the previous word w
	14	a word of length l and the next word w

Global Features

- **Sentence confidence score (*S_{crf}*)**
  - Calculated by CRF++ (tookit by Taku Kudo)
  - E.g. 0.95 for candidate y_1
    - 我们(we) / 生活(live) / 在(in) / 信息(information) / 时代(age)

- **Sentence language model score (*S_{lm}*)**
  - Produced by SRILM (Stolcke, 2002) toolkit, in log-probability format
  - E.g. -10 for candidate y_1
  - Normalization:
    - abs(S_{lm} / sentence_length) = | -10 / 5 | = 2

Experimental Data Sets

- Three corpora from the third SIGHAN Bakeoff, word segmentation shared task:
  - CityU corpus, MSRA corpus, and UPUC corpus

	CityU	MSRA	UPUC
Number of sentences in Training Set	57,275	46,364	18,804
Number of sentences in Test Set	7,511	4,365	5,117

- PU corpus from the first SIGHAN Bakeoff, word segmentation shared task

	PU
Number of sentences in Training Set	19,056
Number of sentences in Test Set	1,944

Learning “Global Features” Weights

Learning “Global Features” Weights

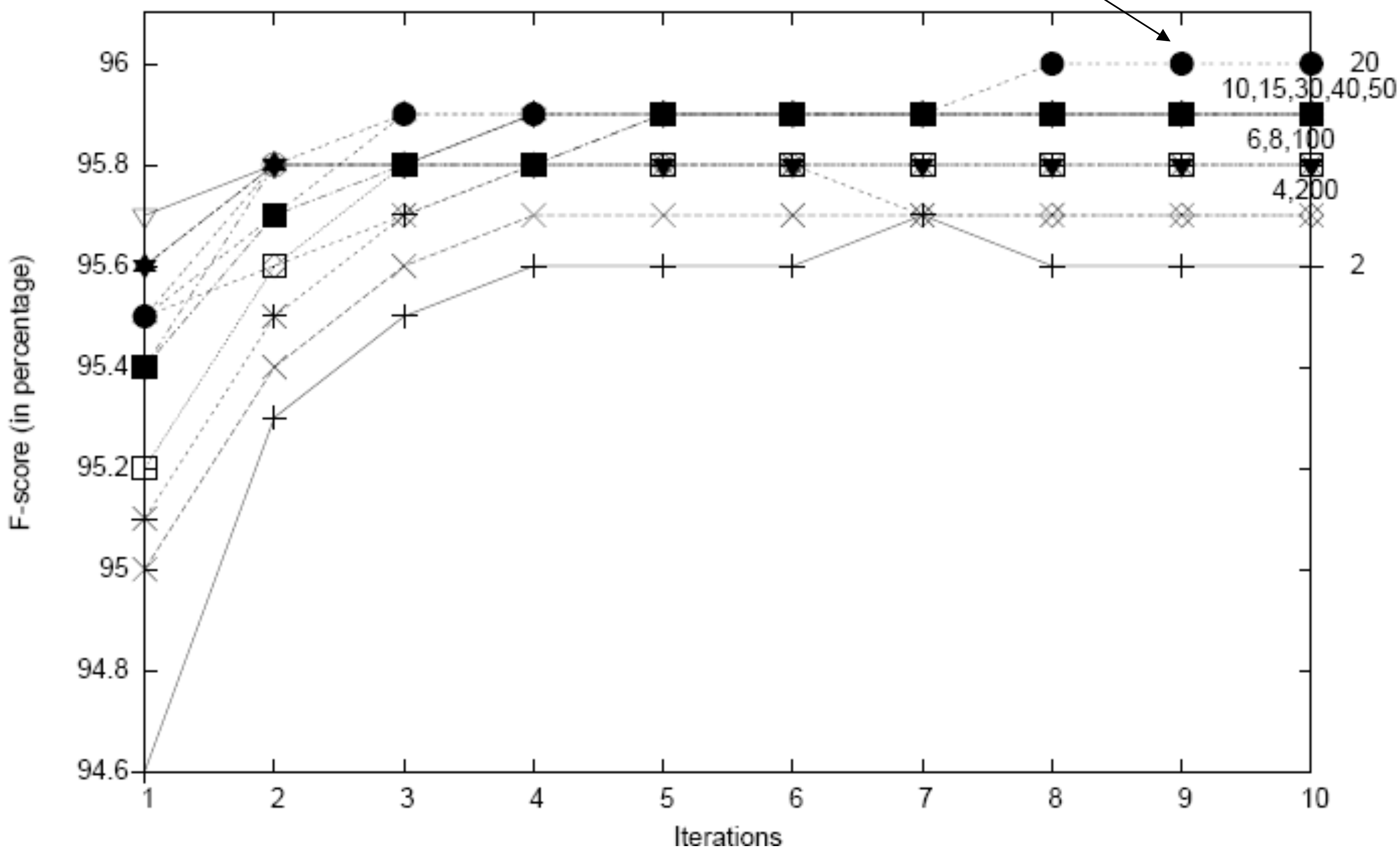
- Compare two options in learning “global feature” weights
  - Fixing weights using a dev. (development) set
    - Scaling
    - Decomposition
  - Training transformed real-valued weights

Fixing weights for "global features"

- For each corpus, weights for S_{crf} and for S_{lm} are determined using a dev. set and are fixed during training
  - Training set (80%), dev. set (20%)
  - 12 weight values are tested:
    - 2, 4, 6, 8, 10, 15, 20, 30, 40, 50, 100, 200
  - 12 x 12 = 144 combinations of different weight values
    - Assume weights for both "global features" are identical.
    - Assumption based on the fact that weights for these "global features" simply provide an important factor
      - only a threshold is needed rather than a finely tuned value

Learning Global Features Weights from Development Data

W=20 gives the highest score



UPUC Development Set

Training transformed real-valued weights

- (Liang, 2005) incorporated and learned weights for real-valued mutual information (MI) features by transforming them into alternative forms:
  - Scale values from $[0, \infty)$ into some fixed range $[a, b]$
    - smallest value observed maps to a
    - largest value observed maps to b
  - Apply z-scores instead of the original values. The z-score of value x from $[0, \infty)$ is $(x-\mu)/\sigma$, where μ and σ represent the mean and the standard deviation of the distribution of x values
  - Map any value x to a if $x < \mu$, the mean value from the distribution of x values, or to b if $x \geq \mu$

Training transformed real-valued weights with averaged perceptron

Method	F-score (UPUC)		F-score (CityU)	
	held-out set	test set	held-out set	test set
Without "global features"	95.5	92.5	97.3	96.7
Fix "global feature" weight	**96.0**	**93.1**	**97.7**	**97.1**
Threshold at mean to 0, 1	95.0	92.0	96.7	96.0
Threshold at mean to -1, 1	95.0	92.0	96.7	96.0
Normalize to [0, 1]	95.2	92.1	96.8	96.0
Normalize to [-1, 1]	95.1	92.0	96.8	95.9
Normalize to [-3, 3]	95.1	92.1	96.8	96.0
Z-score	95.4	92.5	97.1	96.3

- Z-scores perform well but do not out-perform fixing "global feature" weights using the development set.
  - The two "global features" do not have shared components across different training sentences

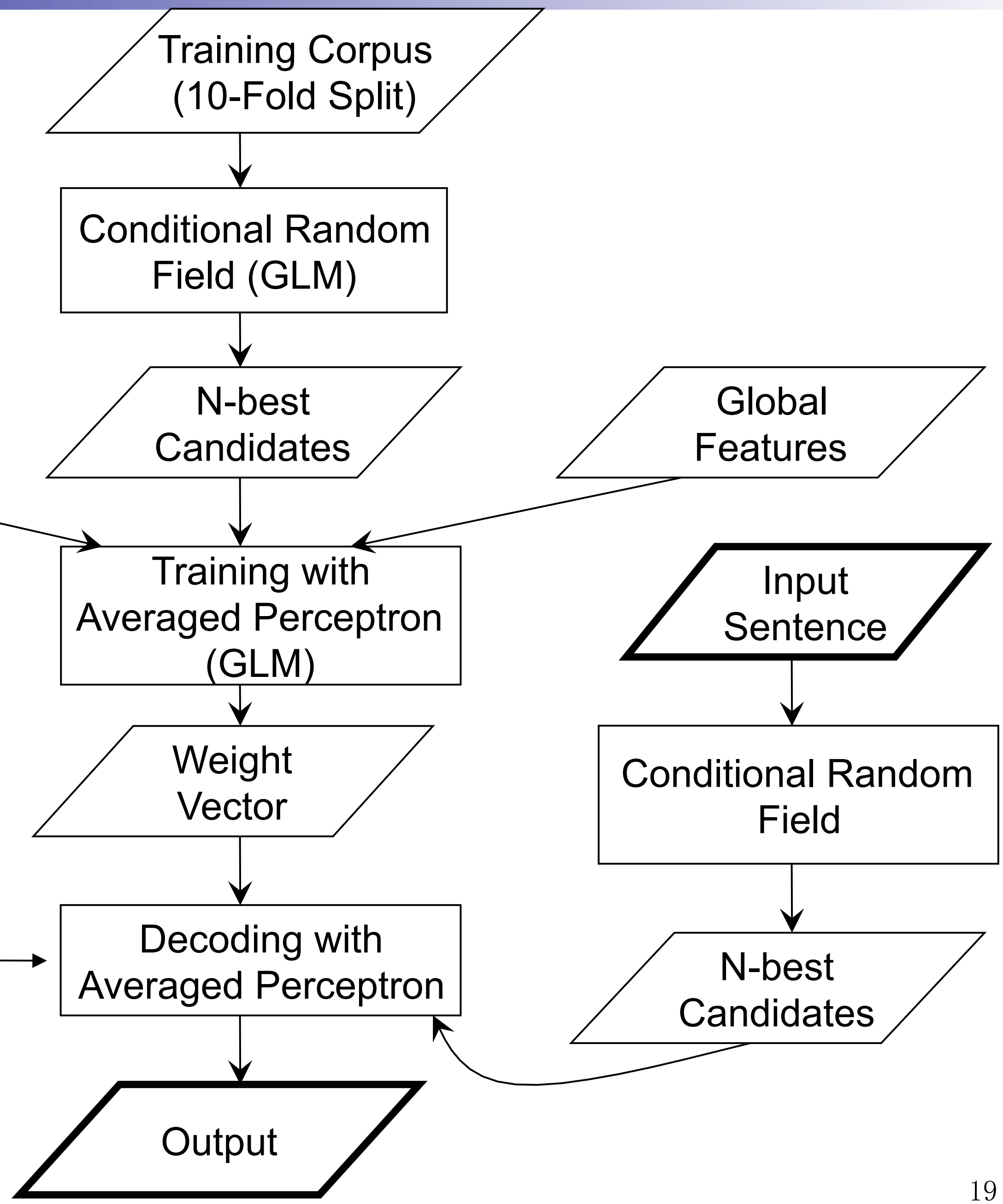
Re-ranking vs. Beam Search

Re-ranking vs. Beam Search

$$y' = \arg\max_{\underbrace{y \in GEN(x)}} \left(\sum_k w_k \cdot f_k(x, y) \right)$$

- Re-ranking with a finite number of candidates
  - E.g. 100 best candidates from another system
- Using all possible segmentations
  - *Dynamic programming*, used when every sub-segmentation has a probability score
  - *Beam search*, when training method uses mistake-driven updates

Re-ranking with Averaged Perceptron

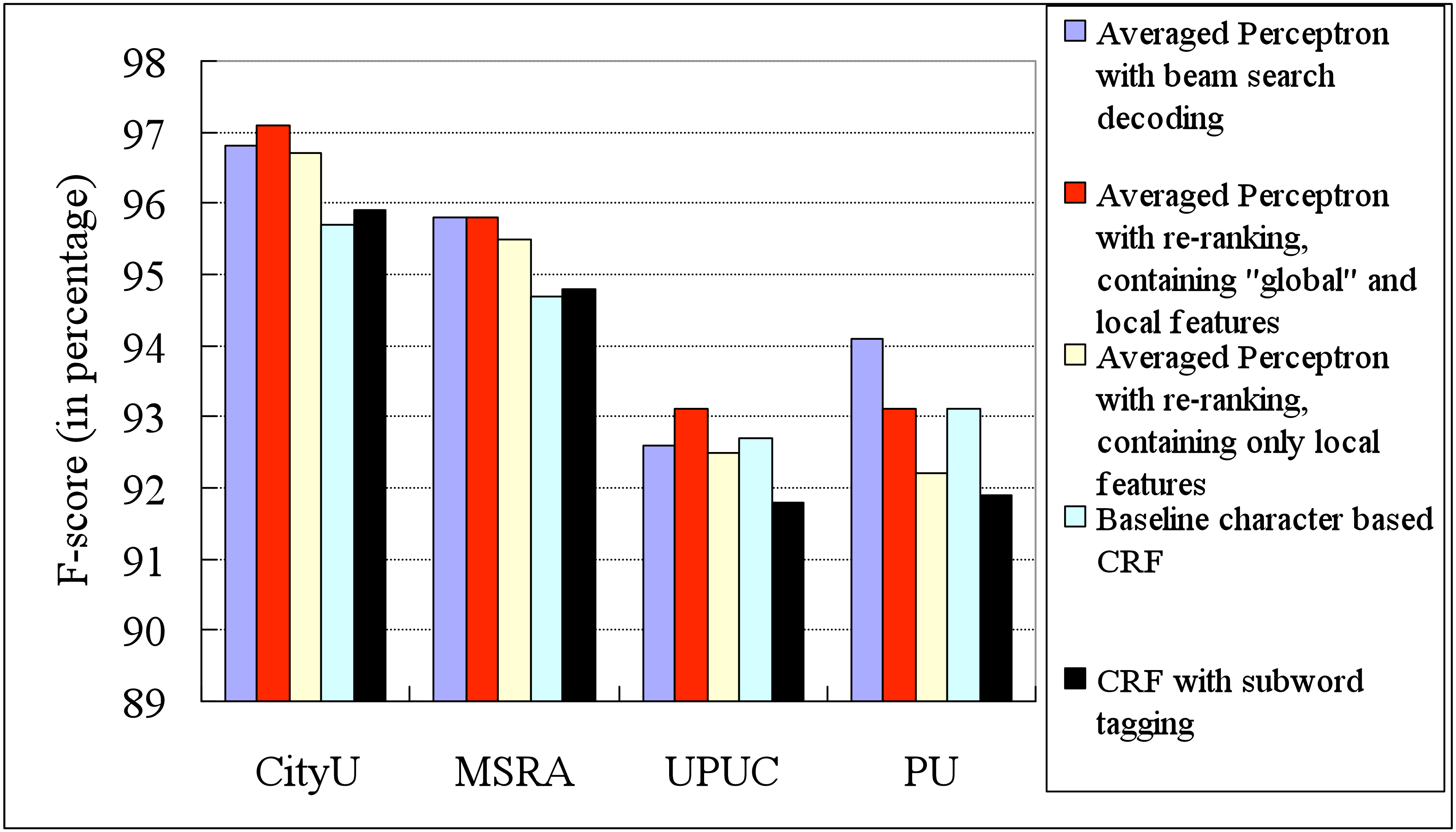


$$\arg\max_{y \in GEN(x)} \left(\sum_k w_k \cdot f_k(x, y)\right)$$

Beam Search

- Beam Search Decoding:
  - Zhang (Collins and Roark, 2004; Zhang and Clark, 2007) proposed beam search decoding using only local features
  - We implemented beam search decoding for averaged perceptron
  - This decoder reads characters from input one at a time, and generates candidate segmentations incrementally.
    - At each stage, the next character is
      - Either appended to the last word in the candidate
      - Or taken as the start of a new word
    - Only maximum B best candidates are retained in each stage
    - After last character is processed, the decoder returns the candidate with the best score.

Re-ranking vs. Beam Search



- (Test set) Compare the truth with 20-best list to see whether the gold standard is in this 20-best list CRF++ produced:

CityU	MSRA	UPUC	PU
88.2%	88.3%	68.4%	54.8%

Averaged Perceptron vs. Max-Margin (EG)

Averaged Perceptron vs. Max-Margin (EG)

- **Perceptron:** Accuracy depends on the margin in the data, but doesn't maximize the margin
- **EG (Exponentiated Gradient) algorithm**
  - Explicitly maximizes the margin M between the truth and the candidates. M is defined as

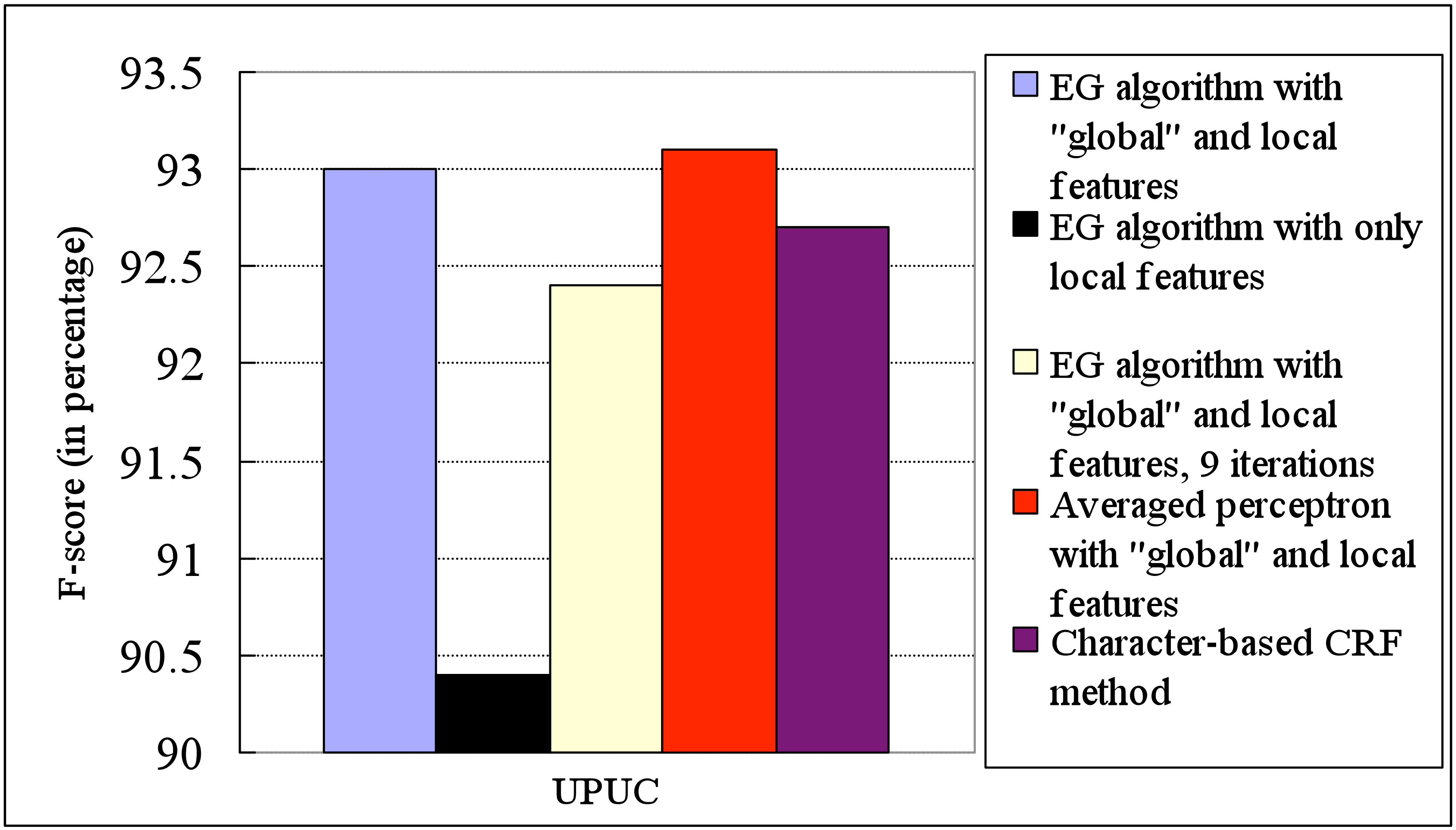
$$M_{i,y} = \underbrace{\overline{f}(x_i, y_i) \bullet \overline{w}}_{\text{Truth}} - \underbrace{\overline{f}(x_i, y) \bullet \overline{w}}_{\text{Candidate}}$$

and $\overline{w}$ is calculated as

$$\overline{w} = \sum_{i,y} \alpha_{i,y} [\overline{f}(x_i, y_i) - \overline{f}(x_i, y)]$$

Averaged Perceptron vs. EG Algorithm

In EG, weights for global features are set to 90, and iteration T = 22, on UPUC



Summary

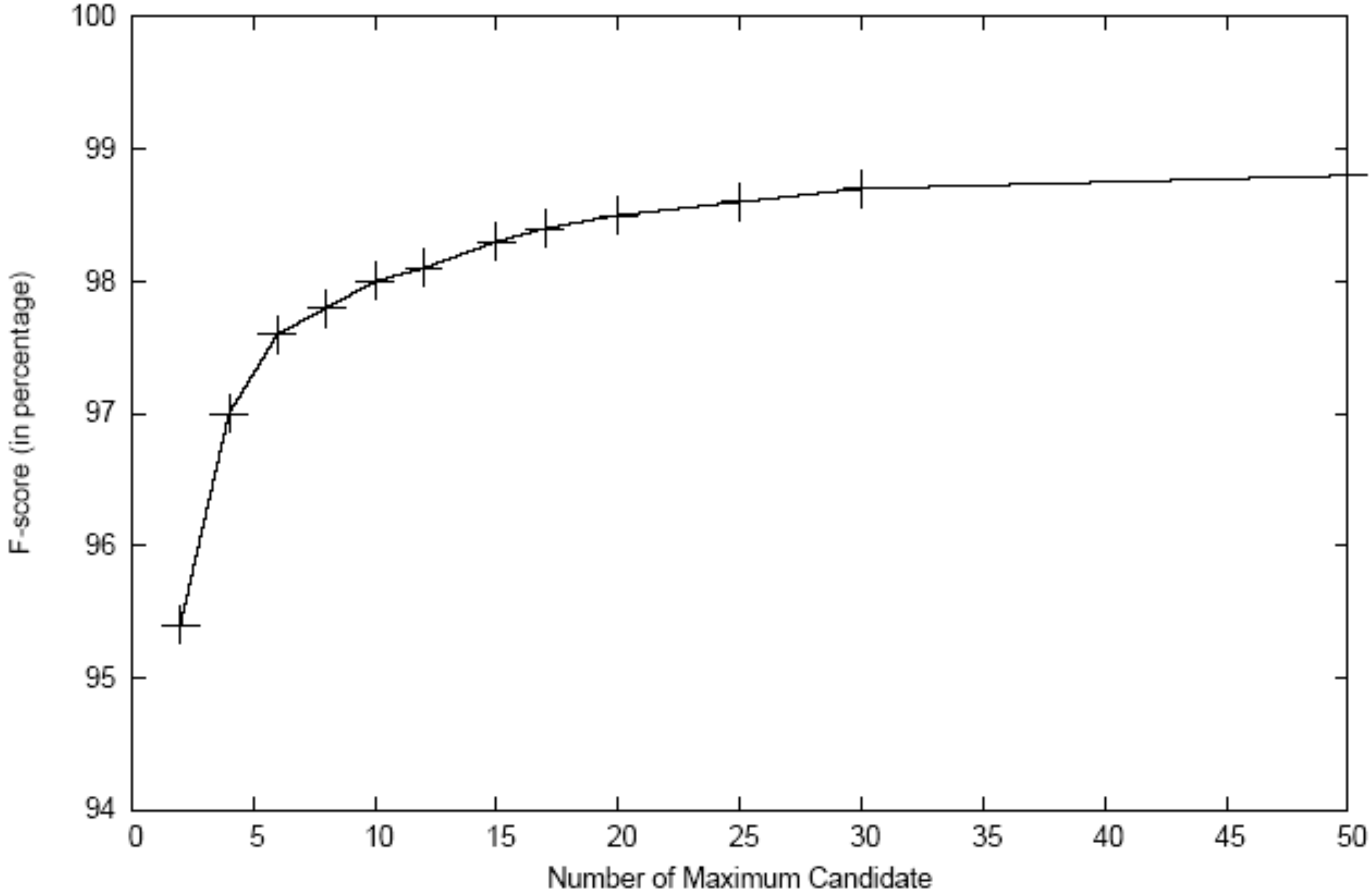
- Explored several choices in building a Chinese word segmentation system:
  - Found that using a development dataset to fix these feature weights is better than learning them from data directly
  - Compared re-ranking versus the use of full beam search decoding, and found that better engineering is required to make beam search competitive in all datasets
  - Explored the choice between a max-margin global linear model and an averaged perceptron global linear model, and found that the averaged perceptron is typically faster and as accurate for our datasets.

Future Work

- Applying N-best re-ranking into rescoring beam search results
- Incorporating the sentence language model score "global feature" into beam search
  - Cube pruning (Huang and Chiang, 2007)
- Better Engineering
  - EG is computational expensive since it requires more iterations to maximize the margin; therefore, we only tested on UPUC corpus.
  - However, the baseline CRF model performs quite well on UPUC
  - In order to compare EG in other larger corpora, better engineering is desired for faster computing

Thank you!

Re-ranking vs. Beam Search



- To balance accuracy and speed, n = 20

Re-ranking vs. Beam Search

- Weight for the *sentence confidence score* (S_{crf}) feature and that for the *sentence language model score* (S_{lm}) feature, and training iterations, are chosen to be:

	CityU	MSRA	UPUC	PU
Weight for S_{crf} and S_{lm}	15	15	20	40
Training Iterations	7	7	9	6

Re-ranking vs. Beam Search

	CityU	MSRA	UPUC	PU
Beam Size	16	16	16	16
Training Iterations	7	7	9	6

Significance Test (McNemar's Test)

Data Set	**P-Value**
CityU	≤ 2.04e-319
MSRA	≤ 7e-74
UPUC	≤ 2.5e-25

Averaged Perceptron vs. EG Algorithm

EG (Exponentiated Gradient) algorithm

- Converges to the minimum of:

$$\sum_i \max_y (1 - M_{i,y})_+ + \frac{1}{2}\|\bar{w}\|^2$$

where

$$(1 - M_{i,y})_+ = \begin{cases} (1 - M_{i,y}) & \text{if } (1 - M_{i,y}) > 0 \\ 0 & \text{otherwise} \end{cases}$$

- In dual optimization representation, choosing **α** values to maximize the dual objective:

$$Q(\bar{\alpha}) = \sum_{i, y \neq y_i} \alpha_{i,y} - \frac{1}{2}\|\bar{w}\|^2$$

EG Algorithm Convergence on UPUC Corpus