Knowledge Integration Into Language Models: A Random Forest Approach

Yi Su

Dissertation Defense

Committee: Prof. Frederick Jelinek, Prof. Sanjeev Khudanpur (Readers)
And Prof. Gerard G. L. Meyer

Department of Electrical and Computer Engineering
The Johns Hopkins University

March 9, 2009

Outline

Outline

What Is A Language Model?

- How likely will a sentence be uttered by a human?
- Complete the sentence:

Wouldn't it be . . .

- . . . great?
- . . . awesome?
- . . . lovely?
- . . . loverly!

Lots of choc'lates for me to eat,
Lots of coal makin' lots of 'eat.
Warm face, warm 'ands, warm feet,
Aow, wouldn't it be loverly?

What Is A Language Model?

- How likely will a sentence be uttered by a human?
- Complete the sentence:

Wouldn't it be . . .

- . . . great?
- . . . awesome?
- . . . lovely?
- . . . loverly!

Lots of choc'lates for me to eat,
Lots of coal makin' lots of 'eat.
Warm face, warm 'ands, warm feet,
Aow, wouldn't it be loverly?

What Is A Language Model?

- How likely will a sentence be uttered by a human?
- Complete the sentence:

Wouldn't it be . . .

- . . . great?
- . . . awesome?
- . . . lovely?
- . . . loverly!

Lots of choc'lates for me to eat,
Lots of coal makin' lots of 'eat.
Warm face, warm 'ands, warm feet,
Aow, wouldn't it be loverly?

What Is A Language Model?

- How likely will a sentence be uttered by a human?
- Complete the sentence:

Wouldn't it be . . .

- . . . great?
- . . . awesome?
- . . . lovely?
- . . . loverly!

Lots of choc'lates for me to eat,
Lots of coal makin' lots of 'eat.
Warm face, warm 'ands, warm feet,
Aow, wouldn't it be loverly?

What Is A Language Model?

- How likely will a sentence be uttered by a human?
- Complete the sentence:

Wouldn't it be . . .

- . . . great?
- . . . awesome?
- . . . lovely?
- . . . loverly!

Lots of choc'lates for me to eat,
Lots of coal makin' lots of 'eat.
Warm face, warm 'ands, warm feet,
Aow, wouldn't it be loverly?

What Is A Language Model?

- How likely will a sentence be uttered by a human?
- Complete the sentence:

Wouldn't it be . . .

- . . . great?
- . . . awesome?
- . . . lovely?
- . . . loverly!

Lots of choc'lates for me to eat,
Lots of coal makin' lots of 'eat.
Warm face, warm 'ands, warm feet,
Aow, wouldn't it be loverly?

State of the Art

- *N*-gram language models remain the *de facto* standard
  - Ignore the fact that we are modeling human language
- But we know so much more about language!
  - give, gave, given (morphology)
  - love (verb), lover (noun), lovely (adjective) (part-of-speech)
  - this:is::these:are (agreement)
  - ...
- Even machines "know" something
  - Morphological analyzers
  - Part-Of-Speech (POS) taggers
  - Parsers
  - ...
- Putting language into language modeling (Jelinek and Chelba, 1999)

State of the Art

- *N*-gram language models remain the *de facto* standard
  - Ignore the fact that we are modeling human language
- But we know so much more about language!
  - give, gave, given (morphology)
  - love (verb), lover (noun), lovely (adjective) (part-of-speech)
  - this:is::these:are (agreement)
  - ...
- Even machines "know" something
  - Morphological analyzers
  - Part-Of-Speech (POS) taggers
  - Parsers
  - ...
- Putting language into language modeling (Jelinek and Chelba, 1999)

Outline

Language Models (LMs)

- A probability distribution over all possible word sequences $P(W)$, where $W = w_1 \dots w_N \in V^*$, V is the vocabulary.
- Decompose using the chain rule

$$P(W) = \prod_{i=1}^{N} P(w_i \mid w_1, \dots, w_{i-1}) \approx \prod_{i=1}^{N} P(w_i \mid \Phi(w_1, \dots, w_{i-1})),$$

where $\Phi : V^* \mapsto C$ is an equivalence mapping of histories.

- An important component in speech recognition, machine translation and information retrieval system.

Decision Tree Language Models

- Language modeling as equivalence mapping of histories
- *N*-gram language models
  - Markovian assumption

$$P(w \mid h) \approx P(w \mid \Phi(h)) = P(w \mid w_{i-n+1}^{i-1}),$$

where $h = w_1, \ldots, w_{i-1} = w_1^{i-1}$.

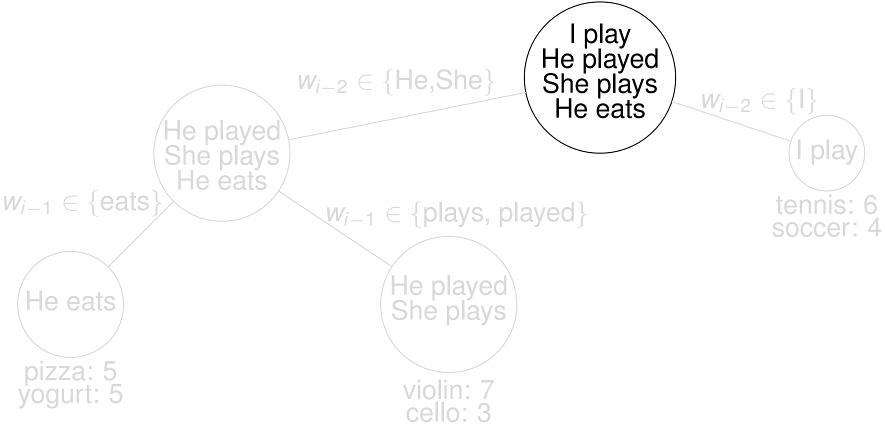
- Decision tree language models (Bahl et al., 1989)
  - Decision tree classifier as equivalence mapping

$$P(w \mid h) \approx P(w \mid \Phi(h)) = P(w \mid \Phi_{DT}(h)).$$

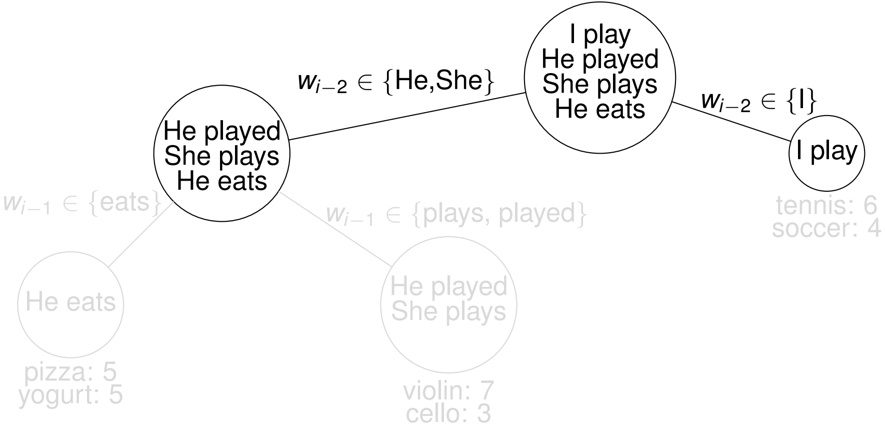
Decision Tree Training

- Growing (Top-down)
  - Start from the root node, which contains all *n*-gram histories in the training text;
  - Recursively split every node to increase the likelihood of the training text by an exchange algorithm (Martin, Liermann and Ney, 1998);
  - Until splitting can no longer increase the likelihood.
- Pruning (Bottom-up)
  - Define the potential of a node as the gain in heldout text likelihood by growing it into a sub-tree
  - Prune away nodes whose potentials fall below a threshold.

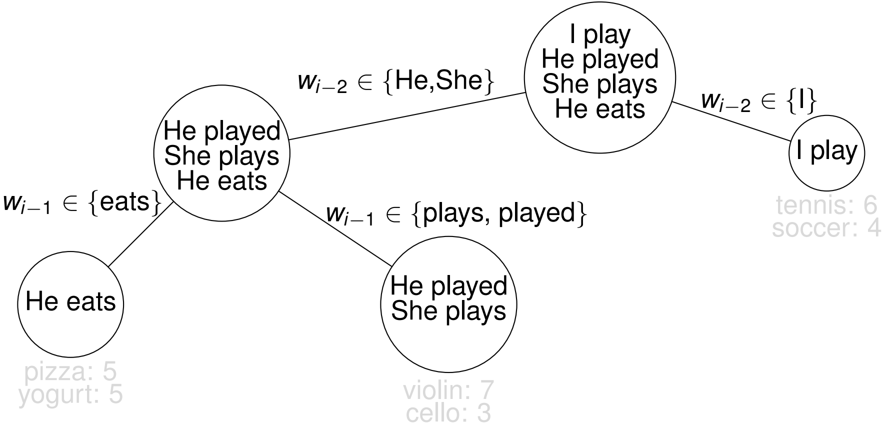
Decision Tree Language Models: Training



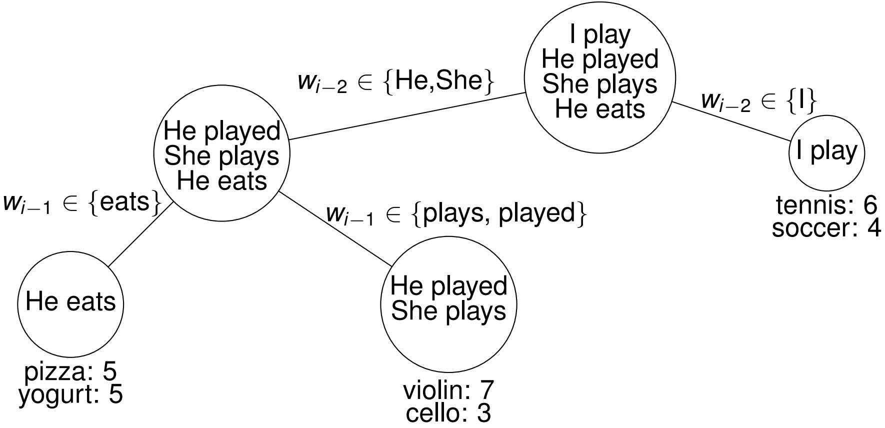
Decision Tree Language Models: Training



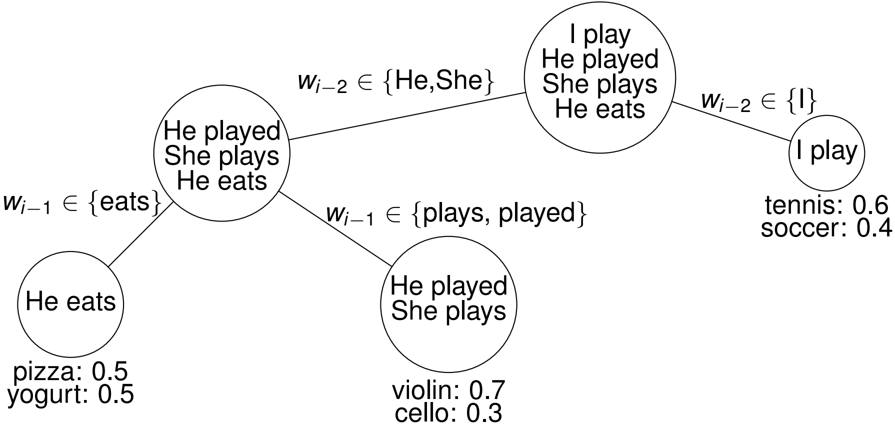
Decision Tree Language Models: Training



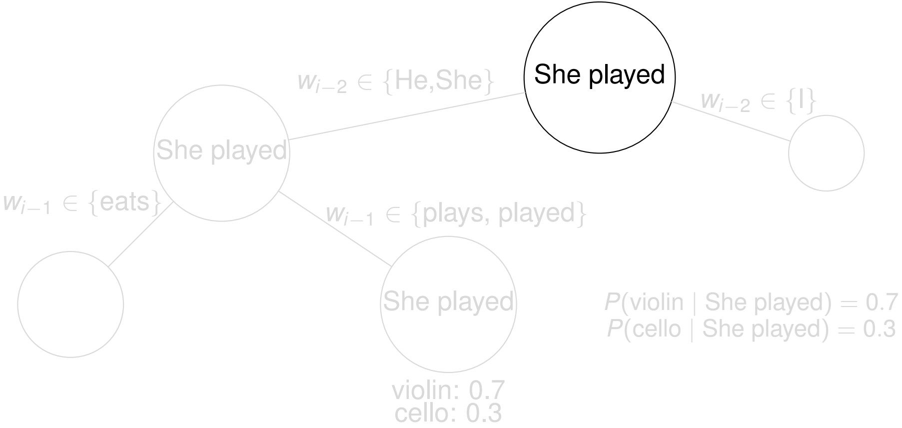
Decision Tree Language Models: Training



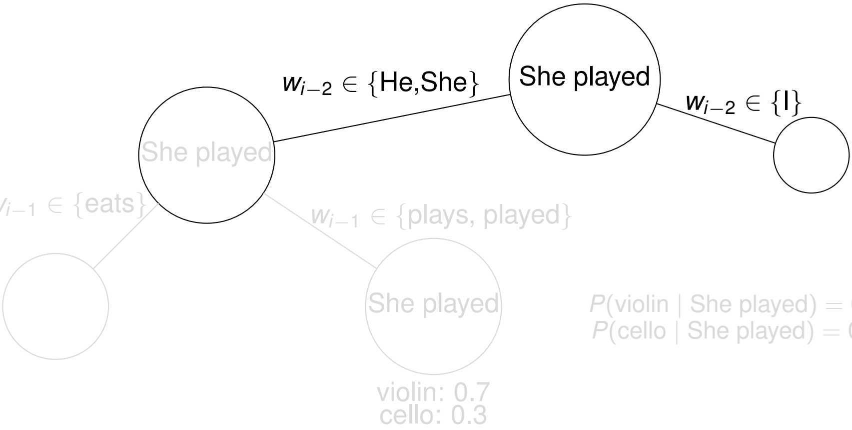
Decision Tree Language Models: Training



Decision Tree Language Models: Testing

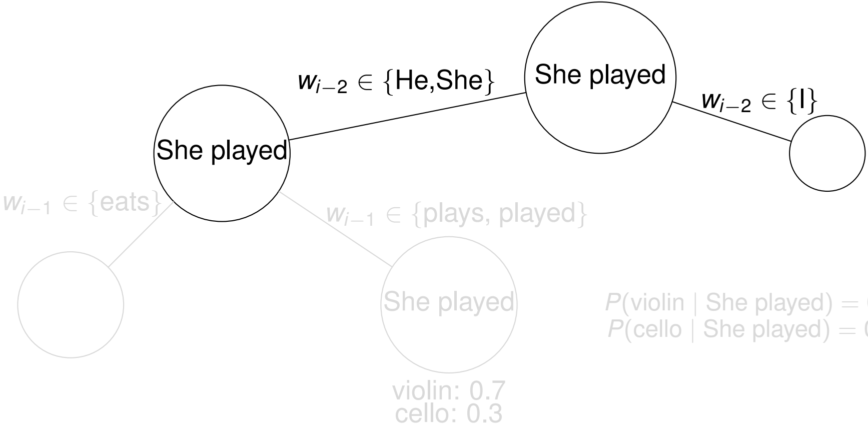


Decision Tree Language Models: Testing



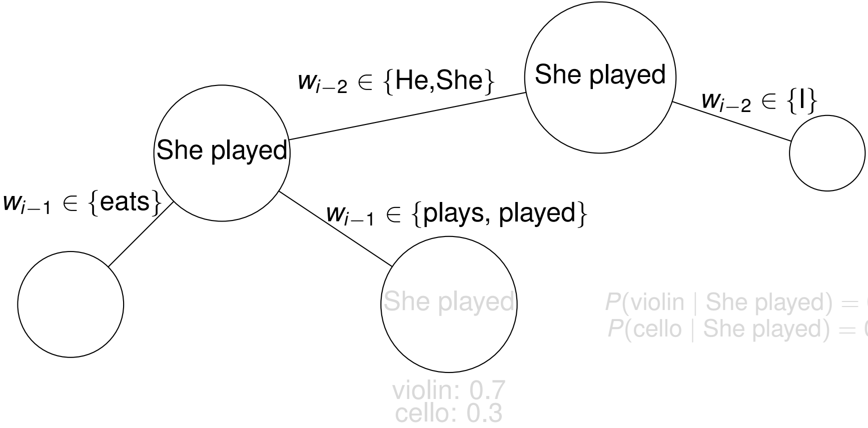
$P(\text{violin} \mid \text{She played}) = 0.7$
$P(\text{cello} \mid \text{She played}) = 0.3$

Decision Tree Language Models: Testing



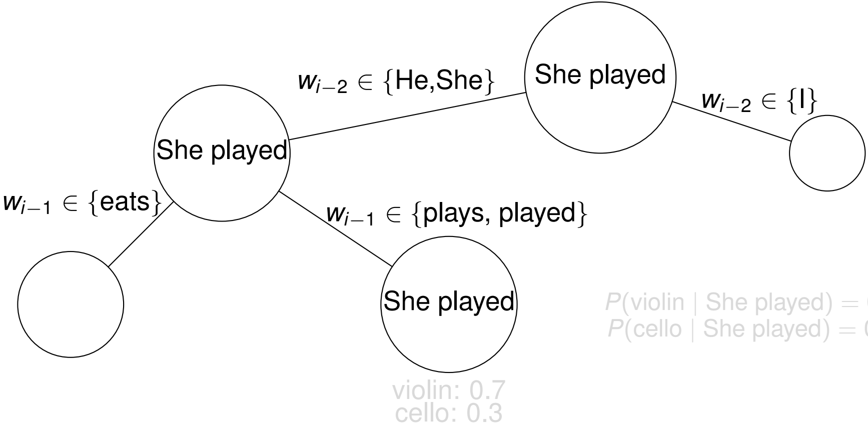
$P(\text{violin} \mid \text{She played}) = 0.7$
$P(\text{cello} \mid \text{She played}) = 0.3$

Decision Tree Language Models: Testing



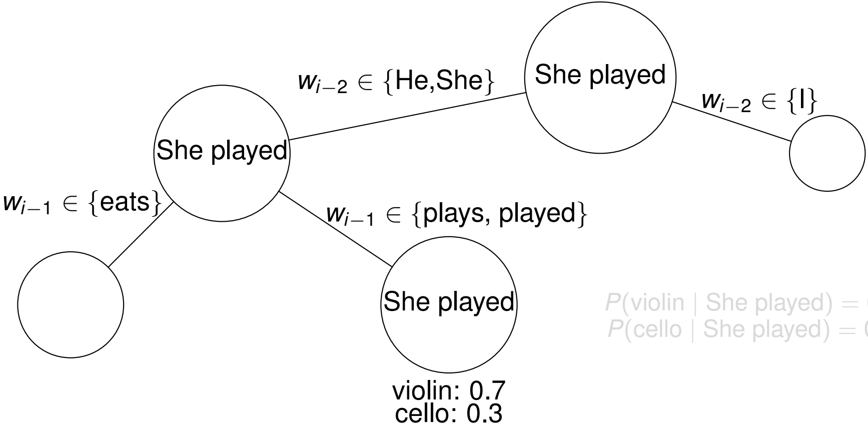
$P(\text{violin} \mid \text{She played}) = 0.7$
$P(\text{cello} \mid \text{She played}) = 0.3$

Decision Tree Language Models: Testing



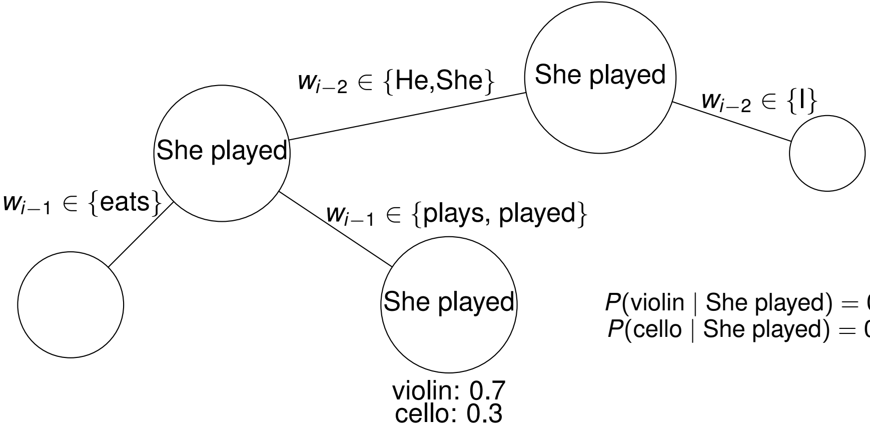
$P(\text{violin} \mid \text{She played}) = 0.7$
$P(\text{cello} \mid \text{She played}) = 0.3$

Decision Tree Language Models: Testing



$P(\text{violin} \mid \text{She played}) = 0.7$
$P(\text{cello} \mid \text{She played}) = 0.3$

Decision Tree Language Models: Testing



$$P(\text{violin} \mid \text{She played}) = 0.7$$
$$P(\text{cello} \mid \text{She played}) = 0.3$$

Decision Tree Language Models

- Failed to improve upon *n*-gram language models (Potamianos and Jelinek,1998)
  - Without efficient search algorithm, greedy tree building can't find a good tree
  - Failed to control the variance
- Random forest (Breiman, 2001)
  - A collection of randomized decision trees
  - Reach final decision by voting to reduce variance
  - Good results in many classification tasks

Decision Tree Language Models

- Failed to improve upon *n*-gram language models (Potamianos and Jelinek,1998)
  - Without efficient search algorithm, greedy tree building can't find a good tree
  - Failed to control the variance
- Random forest (Breiman, 2001)
  - A collection of randomized decision trees
  - Reach final decision by voting to reduce variance
  - Good results in many classification tasks

Outline

Random Forest Language Models (RFLMs)

- A collection of randomized decision tree language models or an i.i.d. sample of decision trees (Xu and Jelinek, 2004)
- Probability via averaging

$$P(w \mid h) = \frac{1}{M} \sum_{j=1}^{M} P(w \mid \Phi_{DT_j}(h))$$

- Superior to *n*-gram language model in terms of perplexity and word error rate on small size corpora (Xu and Mangu, 2005)

Training Randomization

- Random initialization of the exchange algorithm
  - Combat local maximum problem caused by greediness of the exchange algorithm (Martin, Liermann and Ney, 1998)
- Random selection of questions
  - Set membership of a word in a history position j

$$q_S^j(w_1^{i-1}) = \begin{cases} 1 & \text{, if } w_j \in S; \\ 0 & \text{, otherwise.} \end{cases}$$

    where $1 \leq j \leq i-1$ and $S \subset V$.
  - Randomly choose a subset of history positions to investigate
- Random sampling of the training data

Smoothing RFLM

- Kneser-Ney-style smoothing

$$
\begin{aligned}
P(w_i \mid w_{i-n+1}^{i-1}) &= \frac{\max(C(w_i, \Phi(w_{i-n+1}^{i-1})) - D, 0)}{C(\Phi(w_{i-n+1}^{i-1}))} \\
&+ \lambda(\Phi(w_{i-n+1}^{i-1})) P_{KN}(w_i \mid w_{i-n+2}^{i-1})
\end{aligned}
$$

- Can be improved by modified Kneser-Ney smoothing (Chen and Goodman, 1999)
    - Used in all experiments henceforth.

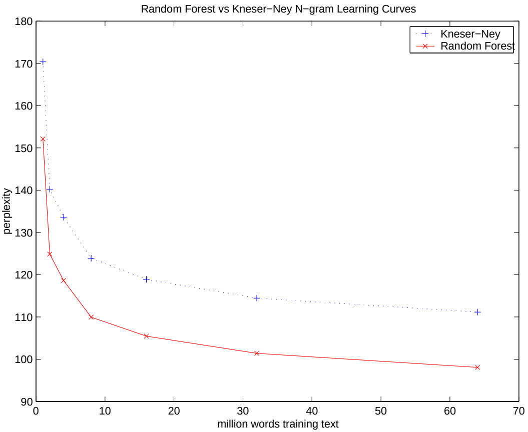
Why *N*-gram LMs Work

- “There is no data like more data.” — Robert L. Mercer
  - Performance of a statistical model depends on the amount of training data
- Simplicity implies scalability
  - *N*-gram LMs outperform complex LMs by using more data

Large-Scale Training and Testing

- Problem: straightforward implementation quickly uses up addressable space.
  - Memory requirement grows as tree grows
- Solution: an efficient disk swapping algorithm exploiting
  - Recursive structure of binary decision tree
    - Compact representation for fast reading and writing
  - Local access property of tree-growing algorithm
    - Node-splitting depends only on the data it contains
- Achieve I/O overhead linear to the size of training *n*-gram types (Su, Jelinek and Khudanpur, 2007).

Learning Curves



Automatic Speech Recognition (ASR)

- System: IBM GALE Mandarin ASR
- Vocabulary: 106K words
- Data: $100M * 7 = 700M$ words for training, 10M for held-out, 20k for testing
- Parameters: 4-grams, 50 trees per forest

Table: Lattice rescoring for IBM GALE Mandarin ASR

Character Error Rate (%)	All	BN	BC
Baseline	18.9	14.2	24.8
RFLM	18.3	13.4	24.4

Outline

1. Introduction
2. Basic Language Models
3. Random Forest Language Models
4. Knowledge Integration with RFLMs
5. Exploiting Prosodic Breaks in LMs
   - Introduction
   - Prosodic Language Models
   - Experimental Results
6. Conclusions

Knowledge Integration

- RFLM as a framework for integrating linguistic knowledge
  - Decision tree can ask any question about the history
- Feature: a function $f(h)$ that maps h to an element of a finite set.

$$f : V^* \mapsto E,$$

  where V is the vocabulary, E is the set of feature values.
- Question: the indicator function $q_S^f(h)$ of the set $f^{-1}(S) = \{h : f(h) \in S \subset E\}$.

$$q_S^f(h) = \begin{cases} 1 & \text{, if } f(h) \in S \subset E; \\ 0 & \text{, otherwise.} \end{cases}$$

Feature Engineering

- Features we have used so far:
  - Word features: if $h_i = w_1 \cdots w_{i-1}$, then

$$\mathrm{WORD}_j(h_i) \doteq w_{i-j},$$

- Features we can potentially use:
  - Any discrete-valued function on the history!
  - E.g., Part-Of-Speech (POS) features: $\mathrm{POS}_j(h_i) \doteq r_{i-j}$, where r_{i-j} is the POS tag of the word w_{i-j}, as provided by an incremental POS tagger.
  - Feature vector representation of a history h

$$F(h) \doteq (f_0(h), f_1(h), \cdots, f_k(h)).$$

Outline

What Is Prosody?

- Suprasegmental properties of spoken language units
- A *wide* range: tone, intonation, stress, break, etc.
- Many applications
  - Disfluency & sentence boundary detection (Stolcke et al, 1998)
  - Topic segmentation (Hirschberg and Nakatani, 1998)
  - Spoken language parsing (Hale et al, 2006)
  - ...
- We are interested in using prosodic breaks for language modeling.

What Is A Prosodic Break Index?

- Number representing subjective strength of one word's association with the next
- On a scale from 0 (the strongest conjoining) to 4 (the most disjoining)
- Example:

Time flies like an arrow.
Time/3 flies/2 like/1 an/0 arrow/4.
Time/1 flies/3 like/2 an/0 arrow/4.

- Prosodic breaks help resolve syntactic ambiguity (Dreyer and Shafran, 2007)
- We think they should help resolve lexical ambiguity, too.

What Is A Prosodic Break Index?

- Number representing subjective strength of one word's association with the next
- On a scale from 0 (the strongest conjoining) to 4 (the most disjoining)
- Example:

Time flies like an arrow.
Time/3 flies/2 like/1 an/0 arrow/4.
Time/1 flies/3 like/2 an/0 arrow/4.

- Prosodic breaks help resolve syntactic ambiguity (Dreyer and Shafran, 2007)
- We think they should help resolve lexical ambiguity, too.

What Is A Prosodic Break Index?

- Number representing subjective strength of one word's association with the next
- On a scale from 0 (the strongest conjoining) to 4 (the most disjoining)
- Example:

Time flies like an arrow.
Time/3 flies/2 like/1 an/0 arrow/4.
Time/1 flies/3 like/2 an/0 arrow/4.

- Prosodic breaks help resolve syntactic ambiguity (Dreyer and Shafran, 2007)
- We think they should help resolve lexical ambiguity, too.

What Is A Prosodic Break Index?

- Number representing subjective strength of one word's association with the next
- On a scale from 0 (the strongest conjoining) to 4 (the most disjoining)
- Example:

Time flies like an arrow.
Time/3 flies/2 like/1 an/0 arrow/4.
Time/1 flies/3 like/2 an/0 arrow/4.

- Prosodic breaks help resolve syntactic ambiguity (Dreyer and Shafran, 2007)
- We think they should help resolve lexical ambiguity, too.

What Is A Prosodic Break Index?

- Number representing subjective strength of one word's association with the next
- On a scale from 0 (the strongest conjoining) to 4 (the most disjoining)
- Example:

Time flies like an arrow.
Time/3 flies/2 like/1 an/0 arrow/4.
Time/1 flies/3 like/2 an/0 arrow/4.

- Prosodic breaks help resolve syntactic ambiguity (Dreyer and Shafran, 2007)
- We think they should help resolve lexical ambiguity, too.

What Is A Prosodic Break Index?

- Number representing subjective strength of one word's association with the next
- On a scale from 0 (the strongest conjoining) to 4 (the most disjoining)
- Example:

Time flies like an arrow.
Time/3 flies/2 like/1 an/0 arrow/4.
Time/1 flies/3 like/2 an/0 arrow/4.

- Prosodic breaks help resolve syntactic ambiguity (Dreyer and Shafran, 2007)
- We think they should help resolve lexical ambiguity, too.

Speech Recognition with Side Information

- Proposal 1: If S is hidden, then

$$W^* = \arg\max_W P(W \mid A) = \arg\max_W P(A \mid W) \sum_S P(W, S).$$

- Proposal 2: If S is observable, then

$$(W, S)^* = \arg\max_{W,S} P(W, S \mid A) \approx \arg\max_{W,S} P(A \mid W) P(W, S).$$

Are Prosodic Breaks Hidden or Observable?

- Strictly speaking, only acoustic features are observable in speech recognition;
- However, unlike hidden structures such as parse trees, prosodic breaks can be predicted from acoustic features with high precision. (Hale et al, 2006)
  - 83.12% for predicting a 3-valued break on annotated Switchboard
- Each case has its pros and cons.
- We are going to investigate these two options for the purpose of language modeling.

Joint Model of Words and Breaks

$$P(W, S) \approx \prod_{i=0}^{m} P(w_i, s_i \mid w_{i-n+1}^{i-1}, s_{i-n+1}^{i-1})$$

- Tuple Model: Let $t_i = (w_i, s_i)$, for all $0 \leq i \leq m$. We have

$$P(w_i, s_i \mid w_{i-n+1}^{i-1}, s_{i-n+1}^{i-1}) = P(t_i \mid t_{i-n+1}^{i-1}).$$

- Decomposed Model

$$\begin{aligned} & P(w_i, s_i \mid w_{i-n+1}^{i-1}, s_{i-n+1}^{i-1}) \\ = \quad & P(w_i \mid w_{i-n+1}^{i-1}, s_{i-n+1}^{i-1}) P(s_i \mid w_{i-n+1}^{i}, s_{i-n+1}^{i-1}) \end{aligned}$$

One Problem

$$P(s_i \mid w_{i-n+1}^{i}, s_{i-n+1}^{i-1}) = ?$$

- How do we smooth things like this? Back-off! Deleted interpolation!
- In what order do we back off or delete? Well...
  - No "natural order" of backing off
  - Previous research either relied on heuristics (Chelba and Jelinek, 2000; Charniak, 2001)
  - Or tried to find the "optimal" path or combination of paths (Bilmes and Kirchhoff, 2003; Duh and Kirchhoff, 2004)
- We have something better... Random Forests!

One Problem

$$P(s_i \mid w_{i-n+1}^{i}, s_{i-n+1}^{i-1}) = ?$$

- How do we smooth things like this? Back-off! Deleted interpolation!
- In what order do we back off or delete? Well...
  - No “natural order” of backing off
  - Previous research either relied on heuristics (Chelba and Jelinek, 2000; Charniak, 2001)
  - Or tried to find the “optimal” path or combination of paths (Bilmes and Kirchhoff, 2003; Duh and Kirchhoff, 2004)
- We have something better... Random Forests!

One Problem

$$P(s_i \mid w_{i-n+1}^{i}, s_{i-n+1}^{i-1}) = ?$$

- How do we smooth things like this? Back-off! Deleted interpolation!
- In what order do we back off or delete? Well...
  - No "natural order" of backing off
  - Previous research either relied on heuristics (Chelba and Jelinek, 2000; Charniak, 2001)
  - Or tried to find the "optimal" path or combination of paths (Bilmes and Kirchhoff, 2003; Duh and Kirchhoff, 2004)
- We have something better... Random Forests!

One Problem

$$P(s_i \mid w_{i-n+1}^{i}, s_{i-n+1}^{i-1}) = ?$$

- How do we smooth things like this? Back-off! Deleted interpolation!
- In what order do we back off or delete? Well...
  - No “natural order” of backing off
  - Previous research either relied on heuristics (Chelba and Jelinek, 2000; Charniak, 2001)
  - Or tried to find the “optimal” path or combination of paths (Bilmes and Kirchhoff, 2003; Duh and Kirchhoff, 2004)
- We have something better... Random Forests!

One Problem

$$P(s_i \mid w_{i-n+1}^{i}, s_{i-n+1}^{i-1}) = ?$$

- How do we smooth things like this? Back-off! Deleted interpolation!
- In what order do we back off or delete? Well...
  - No “natural order” of backing off
  - Previous research either relied on heuristics (Chelba and Jelinek, 2000; Charniak, 2001)
  - Or tried to find the “optimal” path or combination of paths (Bilmes and Kirchhoff, 2003; Duh and Kirchhoff, 2004)
- We have something better... Random Forests!

One Problem

$$P(s_i \mid w_{i-n+1}^{i}, s_{i-n+1}^{i-1}) = ?$$

- How do we smooth things like this? Back-off! Deleted interpolation!
- In what order do we back off or delete? Well...
  - No “natural order” of backing off
  - Previous research either relied on heuristics (Chelba and Jelinek, 2000; Charniak, 2001)
  - Or tried to find the “optimal” path or combination of paths (Bilmes and Kirchhoff, 2003; Duh and Kirchhoff, 2004)
- We have something better... Random Forests!

Ask the Right Question

- Questions
  - We have asked:

    Is the word w_{i-1} in the set of words $\{a, an, the\}$?
  - We would like to ask:

    Does the prosodic break s_{i-1} take its value in the set of values $\{1, 2, 3\}$?
- Same algorithms for training and testing
- Natural integration with background *n*-gram LM
- Feature selection on-the-fly!

Ask the Right Question

- Questions
  - We have asked:

    Is the word w_{i-1} in the set of words $\{a, an, the\}$?
  - We would like to ask:

    Does the prosodic break s_{i-1} take its value in the set of values $\{1, 2, 3\}$?
- Same algorithms for training and testing
- Natural integration with background *n*-gram LM
- Feature selection on-the-fly!

Experimental Setup

- Vocabulary: 10k
- Data: ToBI-labeled Switchboard (Ostendorf et al., 2001).
  - 666k words for training
  - 51k words for held-out
  - 49k words for testing
- Parameters:
  - history up to 2 words and 2 breaks (“3-grams”)
  - 100 trees per forest

Granularity

- Granularity of prosodic breaks might be too coarse for LM
- Compared 2-, 3- and 12-valued scheme for

$$P(w_i \mid w_{i-1}, w_{i-2}, s_{i-1}, s_{i-2})$$

Table: *Granularity of Prosodic Breaks*

Model	two-level	three-level	cont.-valued
KN.3gm	66.1	66.1	66.1
RF-100	65.5	65.4	56.2

Main Results

Table: *Main Perplexity Results*

Model	Method	KN	RF
$P(W,S)$	tuple 3gm	358	306
	decomp.	274	251
$P(W)$	tuple 3gm	69.3	67.2
$=\sum_S P(W,S)$	decomp.	66.8	64.2
$P(W)$	word 3gm	66.1	62.3

Main Results

Table: *Main Perplexity Results*

Model	Method	KN	RF
$P(W,S)$	tuple 3gm	358	306
	decomp.	274	251
$P(W)$	tuple 3gm	69.3	67.2
$=\sum_S P(W,S)$	decomp.	66.8	64.2
$P(W)$	word 3gm	66.1	62.3

Outline

Conclusions

- Random forest language model as a general framework
  - For integrating knowledge into language models
- Exploiting prosodic breaks in language modeling with random forests (Su and Jelinek, 2008)
  - Finer grained prosodic break indices are needed.
  - Prosodic breaks should be given to language models.

Acknowledgements

- Frederick Jelinek and thesis committee
- Johns Hopkins: Peng Xu, Bill Byrne, Damianos Karakos, Zak Shafran, Markus Dreyer
- IBM: Lidia Mangu, Yong Qin, Geoff Zweig
- OHSU: Brian Roark, Richard Sproat

- Many thanks to my colleagues in CLSP for generous help and invaluable discussions!

Publications

- **Yi Su** and Frederick Jelinek. Exploiting prosodic breaks in language modeling with random forests. In *Proceedings of Speech Prosody*, pages 91–94, Campinas, Brazil, May 2008.
- Jia Cui, **Yi Su**, Keith Hall, and Frederick Jelinek. Investigating linguistic knowledge in a maximum entropy token-based language model. In *Proceedings of ASRU*, Kyoto, Japan, December 2007.
- **Yi Su**, Frederick Jelinek, and Sanjeev Khudanpur. Large-scale random forest language models for speech recognition. In *Proceedings of INTERSPEECH*, volume 1, pages 598–601, Antwerp, Belgium, 2007.
- Yanli Zheng, Richard Sproat, Liang Gu, Izhak Shafran, Haolang Zhou, **Yi Su**, Daniel Jurafsky, Rebecca Starr, and Su-Youn Yoon. Accent detection and speech recognition for Shanghai-accented Mandarin. In *Proceedings of INTERSPEECH*, pages 217–220, Lisbon, Portugal, September 2005.