Tree-based and Forest-Based Translation

Liang Huang

Joint work with Kevin Knight (ISI), Aravind Joshi (Penn), Haitao Mi and Qun Liu (ICT)

UC Berkeley, Feb 6, 2009
Translation is hard!

zi    zhu     zhong    duan
自助 终端

self help terminal device

help oneself terminating machine

(ATM, “self-service terminal”)
Translation is hard!

小心滑落

Slip carefully
Translation is hard!
Translation is hard!
Translation is hard!
or even...
or even...

clear evidence that MT is used in real life.
How do people translate?

1. understand the source language sentence

2. generate the target language translation

布什 与 沙龙 举行 了 会谈

Bùshí  yu  Shalón  juxíng  le  huìtán

Bush and/with Sharon hold [past.] meeting
How do people translate?

1. understand the source language sentence
2. generate the target language translation

布什 与 沙龙 举行 了 会谈

*Bùshí yu Shálóng juxíng le huìtán*

Bush and/with Sharon hold [past.] meeting
How do people translate?

1. understand the source language sentence

2. generate the target language translation

“Bush held a meeting with Sharon”
How do compilers translate?

1. parse high-level language program into a syntax tree
2. generate intermediate or machine code accordingly

\[ x3 = y + 3; \]
How do compilers translate?

1. parse high-level language program into a syntax tree
2. generate intermediate or machine code accordingly

x3 = y + 3;
How do compilers translate?

1. parse high-level language program into a syntax tree
2. generate intermediate or machine code accordingly

```plaintext
x3 = y + 3;
```

LD     R1,  id2
ADDF   R1,  R1, #3.0  // add float
RTOI   R2,  R1        // real to int
ST     id1, R2
How do compilers translate?

1. parse high-level language program into a syntax tree
2. generate intermediate or machine code accordingly

```plaintext
x3 = y + 3;
```

LD     R1,  id2
ADDF   R1,  R1, #3.0  // add float
RTOI   R2,  R1        // real to int
ST     id1, R2

syntax-directed translation (~1960)
Syntax-Directed Machine Translation

1. parse the source-language sentence into a tree
2. recursively convert it into a target-language sentence

Bush hold and/with Sharon meeting

(Irons 1961; Lewis, Stearns 1968; Aho, Ullman 1972) ==> (Huang, Knight, Joshi 2006)
Syntax-Directed Machine Translation

1. parse the source-language sentence into a tree
2. recursively convert it into a target-language sentence

Bush and/with Sharon hold [past.] meeting

(Irons 1961; Lewis, Stearns 1968; Aho, Ullman 1972) ==> (Huang, Knight, Joshi 2006)
Syntax-Directed Machine Translation

- recursive rewrite by pattern-matching

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
Syntax-Directed Machine Translation

- recursive rewrite by pattern-matching

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
Syntax-Directed Machine Translation

- recursive rewrite by pattern-matching

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
Syntax-Directed Machine Translation

- recursive rewrite by pattern-matching
Syntax-Directed Machine Translation?

- recursively solve unfinished subproblems

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
• recursively solve unfinished subproblems

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
Syntax-Directed Machine Translation?

- recursively solve unfinished subproblems

Bush

with

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
Syntax-Directed Machine Translation?

- recursively solve unfinished subproblems

Bush with

(NPB)

Shālóng

(VPB)

VV

jǔxíng

le

huìtán

VV

jǔxíng

le

→ held

x₁

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
 recursively solve unfinished subproblems

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
Syntax-Directed Machine Translation?

- continue pattern-matching

Bush held NPB with NPB
  | huìtán |
  | Shālóng |
Syntax-Directed Machine Translation?

- continue pattern-matching

Bush held with

NPB | huìtán

a meeting

NPB | Shālóng

Sharon

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
Syntax-Directed Machine Translation?

- continue pattern-matching

Bush held a meeting with Sharon

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
Syntax-Directed Machine Translation?

- continue pattern-matching

Bush held a meeting with Sharon

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
Pros: simple, fast, and expressive

- simple architecture: separate parsing and translation
- efficient linear-time dynamic programming
  - “soft decision” at each node on which rule to use
  - (trivial) depth-first traversal with memoization
- expressive multi-level rules for syntactic divergence
  (beyond CFG)

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
Pros: simple, fast, and expressive

- simple architecture: separate parsing and translation
- efficient linear-time dynamic programming
  - “soft decision” at each node on which rule to use
  - (trivial) depth-first traversal with memoization
- expressive multi-level rules for syntactic divergence (beyond CFG)

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
Pros: simple, fast, and expressive

- simple architecture: separate parsing and translation
- efficient linear-time dynamic programming
  - “soft decision” at each node on which rule to use
  - (trivial) depth-first traversal with memoization
- expressive multi-level rules for syntactic divergence (beyond CFG)

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
Pros: simple, fast, and expressive

- simple architecture: separate parsing and translation
- efficient linear-time dynamic programming
  - “soft decision” at each node on which rule to use
  - (trivial) depth-first traversal with memoization
- expressive multi-level rules for syntactic divergence
  (beyond CFG)

(Huang, Knight, Joshi 2006); rules from (Galley et al., 04)
Cons: Parsing Errors

- ambiguity is a fundamental problem in natural languages
- probably will never have perfect parsers (unlike compiling)
- parsing errors affect translation quality!
Cons: Parsing Errors

- ambiguity is a fundamental problem in natural languages
- probably will never have perfect parsers (unlike compiling)
- parsing errors affect translation quality!

emergency exit
or “safe exports”?
Cons: Parsing Errors

- ambiguity is a fundamental problem in natural languages
- probably will never have perfect parsers (unlike compiling)
- parsing errors affect translation quality!

emergency exit  
or “safe exports”?  

mind your head  
or “meet cautiously”?
I saw her duck.
Exponential Explosion of Ambiguity

I saw her duck.
I saw her duck.
Exponential Explosion of Ambiguity

I saw her duck.

• how about...
  • I saw her duck with a telescope.
  • I saw her duck with a telescope in the garden...
Exponential Explosion of Ambiguity

I saw her duck.

- how about...
  - I saw her duck with a telescope.
  - I saw her duck with a telescope in the garden...
Exponential Explosion of Ambiguity

I saw her duck.

• how about...
  • I saw her duck with a telescope.
  • I saw her duck with a telescope in the garden...

NLP == dealing with ambiguities.
Tackling Ambiguities in Translation
Tackling Ambiguities in Translation

- simplest idea: take top-$k$ trees rather than 1-best parse
  - but only covers tiny fraction of the exponential space
  - and these $k$-best trees are very similar
    - e.g., 50-best trees $\sim$ 5-6 binary ambiguities ($2^5 < 50 < 2^6$)
    - very inefficient to translate on these very similar trees
Tackling Ambiguities in Translation

- simplest idea: take top-k trees rather than 1-best parse
  - but only covers tiny fraction of the exponential space
  - and these k-best trees are very similar
    - e.g., 50-best trees ~ 5-6 binary ambiguities ($2^5 < 50 < 2^6$)
    - very inefficient to translate on these very similar trees
- most ambitious idea: combining parsing and translation
  - start from the input string, rather than 1-best tree
  - essentially considering all trees (search space too big)
Tackling Ambiguities in Translation

- simplest idea: take top-\(k\) trees rather than 1-best parse
  - but only covers tiny fraction of the exponential space
  - and these \(k\)-best trees are very similar
    - e.g., 50-best trees \(\sim\) 5-6 binary ambiguities \((2^5 < 50 < 2^6)\)
    - very inefficient to translate on these very similar trees
- most ambitious idea: combining parsing and translation
  - start from the input string, rather than 1-best tree
  - essentially considering all trees (search space too big)
- our approach: packed forest (poly. encoding of exp. space)
  - almost as fast as 1-best, almost as good as combined
Outline

• Overview: Tree-based Translation

• Forest-based Translation
  • Packed Forest
  • Translation on a Forest
  • Experiments

• Forest-based Rule Extraction
  • Large-scale Experiments
From Lattices to Forests

- common theme: polynomial encoding of exponential space
- forest generalizes “lattice/graph” from finite-state world
  - paths => trees  (in DP: knapsack vs. matrix-chain multiplication)
  - graph => hypergraph;  regular grammar => CFG

(Earley 1970; Billot and Lang 1989)
Packed Forest

- a compact representation of many many parses
- by sharing common sub-derivations
- polynomial-space encoding of exponentially large set

(Klein and Manning, 2001; Huang and Chiang, 2005)
Packed Forest

- A compact representation of many many parses
- By sharing common sub-derivations
- Polynomial-space encoding of exponentially large set

(Klein and Manning, 2001; Huang and Chiang, 2005)
Forest-based Translation

“and” / “with”
Forest-based Translation
Forest-based Translation

pattern-matching on forest
(linear-time in forest size)
Forest-based Translation

pattern-matching on forest
(linear-time in forest size)
Forest-based Translation

pattern-matching on forest
(linear-time in forest size)

IP
---
NP
---
x1:NPB
CC
x2:NPB
---
yǔ
“and”

→ x1 x3 with x2

NP0,3
---
NPB0,1
CC1,2
P1,2
NPB2,3
PP1,3
VPB3,6
VP1,6

Bushí
---
yǔ
“and” / “with”

Shālóng
jǔxíng
le
hùítán

与
沙龙
举行
了
会谈
Forest-based Translation

pattern-matching on forest
(linear-time in forest size)
Forest-based Translation

pattern-matching on forest (linear-time in forest size)

布什 与 沙龙 举行 了 会谈

“and” / “with”
Forest-based Translation

pattern-matching on forest
(linear-time in forest size)
Translation Forest
Translation Forest

"held a meeting"

“Bush”

“Sharon”
“Bush held a meeting with Sharon”

“held a meeting”
The Whole Pipeline

input sentence

parser

parse forest

packed forests

pattern-matching w/ translation rules (exact)

translation forest

integrating language models (cube pruning)

translation+LM forest

Alg. 3

1-best translation

k-best translations

(Huang and Chiang, 2005; 2007; Chiang, 2007)
The Whole Pipeline

input sentence

parser

parse forest

forest pruning

pruned forest

pattern-matching w/ translation rules (exact)

translation forest

integrating language models (cube pruning)

translation+LM forest

Alg. 3

l-best translation

k-best translations

(Huang and Chiang, 2005; 2007; Chiang, 2007)
Parse Forest Pruning

- prune *unpromising* hyperedges
- principled way: inside-outside
  - first compute Viterbi inside $\beta$, outside $\alpha$
  - then $\alpha\beta(e) = \alpha(v) + c(e) + \beta(u) + \beta(w)$
  - cost of best deriv that traverses $e$
  - similar to “expected count” in EM
- prune away hyperedges that have $\alpha\beta(e) - \alpha\beta(\text{TOP}) > p$
  for some threshold $p$

Jonathan Graehl: relatively useless pruning
Small-Scale Experiments

• Chinese-to-English translation
  • on a tree-to-string system similar to (Liu et al, 2006)
• 31k sentences pairs (0.8M Chinese & 0.9M English words)
• GIZA++ aligned
• trigram language model trained on the English side
• dev: NIST 2002 (878 sent.); test: NIST 2005 (1082 sent.)
• Chinese-side parsed by the parser of Xiong et al. (2005)
  • modified to output a forest for each sentence (Huang 2008)
• BLEU score: 1-best baseline: 0.2430 vs. Pharaoh: 0.2297
k-best trees vs. forest-based

1.7 Bleu improvement over 1-best, 0.8 over 30-best, and even faster!

\[ k = \sim 6.1 \times 10^8 \] trees

\[ \sim 2 \times 10^4 \] trees
how often is the \( i \)th-best tree picked by the decoder?

- 32% beyond 100-best
- 20% beyond 1000-best

suggested by Mark Johnson

forest as virtual \( \infty \)-best list
wait a sec... where are the rules from?
wait a sec... where are the rules from?

小心  X  <=>  be careful not to X
wait a sec... where are the rules from?

xiǎoxīn gǒu
小心 狗  <=>  be aware of dog

xiǎoxīn
小心  X  <=>  be careful not to X

 PLEASE
BE AWARE
OF DOG.

Slip carefully
wait a sec... where are the rules from?

小心 VP  <=>  be careful not to VP
小心 NP  <=>  be careful of NP
...

xiǎoxīn  gǒu
小心 狗  <=>  be aware of  dog

xiǎoxīn
小心  X  <=>  be careful not to  X
Outline

- Overview: Tree-based Translation
- Forest-based Translation
- Forest-based Rule Extraction
  - background: tree-based rule extraction (Galley et al., 2004)
  - extension to forest-based
  - large-scale experiments
Where are the rules from?

- source parse tree, target sentence, and alignment
- compute target spans

GHKM - (Galley et al 2004; 2006)
Where are the rules from?

- source parse tree, target sentence, and alignment
- well-formed fragment: contiguous and faithful t-span

GHKM - (Galley et al 2004; 2006)
Where are the rules from?

- source parse tree, target sentence, and alignment
- well-formed fragment: contiguous and faithful t-span

admissible set

GHKM - (Galley et al 2004; 2006)
Where are the rules from?

- source parse tree, target sentence, and alignment
- well-formed fragment: contiguous and faithful t-span

GHKM - (Galley et al 2004; 2006)
Where are the rules from?

- source parse tree, target sentence, and alignment
- well-formed fragment: contiguous and faithful t-span

GHKM - (Galley et al 2004; 2006)
Where are the rules from?

- source parse tree, target sentence, and alignment
- well-formed fragment: contiguous and faithful t-span

GHKM - (Galley et al 2004; 2006)
Forest-based Rule Extraction

- same cut set computation; different fragmentation

also in (Wang, Knight, Marcu, 2007)
Forest-based Rule Extraction

- same cut set computation; different fragmentation

Also in (Wang, Knight, Marcu, 2007)
Forest-based Rule Extraction

- same cut set computation; different fragmentation

also in (Wang, Knight, Marcu, 2007)
Forest-based Rule Extraction

- same cut set computation; different fragmentation

\[ \text{IP}(x_1: \text{NPB}, x_2: \text{VP}) \rightarrow x_1 x_2 \]

also in (Wang, Knight, Marcu, 2007)
Forest-based Rule Extraction

- same admissible set definition; different fragmentation

\[ IP(x_1:\text{NPB} \ x_2:\text{VP}) \rightarrow x_1 \ x_2 \]
Forest-based Rule Extraction

- same admissible set definition; different fragmentation

\[ IP(x_1:NPB, x_2:VP) \rightarrow x_1 x_2 \]
Forest-based Rule Extraction

- same admissible set definition; different fragmentation

\[ \text{IP}(x_1: \text{NPB}\ x_2:\text{VP}) \rightarrow x_1 \ x_2 \]
Forest-based Rule Extraction

- same admissible set definition; different fragmentation

\[ \text{IP}(x_1:NPB \ x_2:VP) \rightarrow x_1 \ x_2 \]

[Diagram of a tree structure with nodes labeled as IP, NP, VP, etc., and edges connecting them with labels like "Bush .. Sharon" and "held .. Sharon." A sentence in English is also shown: "Bush held a meeting with Sharon." There are Chinese characters present, likely representing the Chinese sentence corresponding to the English one.]
Forest-based Rule Extraction

- forest can extract smaller chunks of rules

\[ IP(x_1:\text{NPB} \ x_2:\text{VP}) \rightarrow x_1 \ x_2 \]
Forest-based Rule Extraction

- forest can extract smaller chunks of rules

\[ \text{IP}(x_1:NPB \ x_2:VP) \rightarrow x_1 \ x_2 \]

\[ \text{VP} (x_1:PP \ x_2:VPB) \rightarrow x_2 \ x_1 \]
Forest-based Rule Extraction

- forest can extract smaller chunks of rules
The Forest^2 Pipeline

- **training time**
  - source sentence
    - parser
      - word alignment
        - target sentence
  - aligner
    - I-best/forest
      - rule extractor
        - translation ruleset
The Forest² Pipeline

- **Training time**
  - Source sentence
  - Parser
  - Aligner
  - Target sentence

- **Translation time**
  - Source sentence
  - Parser
  - Pattern-matcher
  - Target sentence

- **Rule extractor**
  - Word alignment
  - Rule extractor
  - Translation ruleset
Forest vs. $k$-best Extraction

1.0 Bleu improvement over 1-best, twice as fast as 30-best extraction

~$10^8$ trees
Forest$^2$

- FBIS: 239k sentence pairs (7M/9M Chinese/English words)
- forest in both extraction and decoding
- forest$^2$ results is 2.5 points better than 1-best$^2$
- and outperforms Hiero (Chiang 2007) by quite a bit

<table>
<thead>
<tr>
<th></th>
<th>1-best tree</th>
<th>forest</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-best tree</td>
<td>0.2560</td>
<td>0.2674</td>
</tr>
<tr>
<td>30-best trees</td>
<td>0.2634</td>
<td>0.2767</td>
</tr>
<tr>
<td>forest</td>
<td>0.2679</td>
<td>0.2816</td>
</tr>
<tr>
<td>Hiero</td>
<td>0.2738</td>
<td></td>
</tr>
</tbody>
</table>
Translation Examples

- **src** Powell said the very important talks with Arafat

- **l-best** Powell said the very important talks with Arafat

- **forest** Powell said his meeting with Arafat is very important

- **hiero** Powell said very important talks with Arafat
Conclusions

• main theme: efficient syntax-directed translation
• forest-based translation
  • forest = “underspecified syntax”: polynomial vs. exponential
  • still fast (with pruning), yet does not commit to 1-best tree
  • translating millions of trees is faster than just on top-k trees
• forest-based rule extraction: improving rule set quality
• very simple idea, but works well in practice
  • significant improvement over 1-best syntax-directed
  • final result outperforms hiero by quite a bit
Forest is your friend in machine translation.

help save the forest.

More “forest-based” algorithms in my thesis (this talk is about Chap. 6).
self-service terminals

carefully slide

http://translate.google.com
self-service terminals  carefully slide

http://translate.google.com
self-service terminals carefully

http://translate.google.com
Larger Decoding Experiments (ACL)

- 2.2M sentence pairs (57M Chinese and 62M English words)
- larger trigram models (1/3 of Xinhua Gigaword)
- also use **bilingual phrases** (BP) as flat translation rules
  - phrases that are consistent with syntactic constituents
- forest enables larger improvement with BP

<table>
<thead>
<tr>
<th></th>
<th>T2S</th>
<th>T2S+BP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-best tree</td>
<td>0.2666</td>
<td>0.2939</td>
</tr>
<tr>
<td>30-best trees</td>
<td>0.2755</td>
<td>0.3084</td>
</tr>
<tr>
<td>forest</td>
<td>0.2839</td>
<td>0.3149</td>
</tr>
<tr>
<td>improvement</td>
<td>1.7</td>
<td>2.1</td>
</tr>
</tbody>
</table>