AI = Learning to Translate
Language, Music, and Creativity

Dekai Wu
dekai@cs.ust.hk http://www.cs.ust.hk/~dekai
HKUST
Human Language Technology Center
Department of Computer Science and Engineering
University of Science and Technology, Hong Kong

Where does it go?	Where does it go?
该放哪儿?	它在哪里去了? ✗
應該擺喺邊呀?	它在哪裡去了? ✗
ये कहाँ जाता है ?	यह कहाँ है? ✗
Bu nereye?	Nereye gidiyor? ✗
إلى أين تذهب ؟	أين يذهب؟
Taruh di mana?	Di mana ia pergi? ✗
어디 넣을까?	가 어디에 있습니까? ✗
Saan mo ilalagay ito?	Saan pumunta? ✗
それはどこに行くのでしょうか？	それはどこに行くのでしょうか？ ✗
Où est-ce que ça va?	Où faut-il aller? ✗
Vart ska det?	Vart tar det vägen? ✗

we all use
different
frames of reference

word alignment

Where is the Secretary of Finance when needed ?

财政 司 有需要 时 在 那里 ?

word alignment

aim
- automatically map tokens between input and output sequences

input
- bisequence (input-output pair of sequences)
- token translation lexicon
- language-independent **bracketing ITG** (only 1 generic nonterminal **A**)

output
- aligned bisequence

Implementation
- biparser

(Wu, IJCAI 1995)

BITG (bracketing ITG)

$$
\begin{array}{llll}
A & \xrightarrow{a} & [A\ A] & \\
A & \xrightarrow{a} & \langle A\ A \rangle & \\
A & \xrightarrow{b_{ij}} & u_i/v_j & \text{for all } i,j \text{ English-Chinese lexical translations} \\
A & \xrightarrow{b_{i\epsilon}} & u_i/\epsilon & \text{for all } i \text{ English vocabulary} \\
A & \xrightarrow{b_{\epsilon j}} & \epsilon/v_j & \text{for all } j \text{ Chinese vocabulary}
\end{array}
$$

word alignment

Where is the Secretary of Finance when needed ?

财政 司 有需要 时 在 那里 ?

word alignment

Where is [the Secretary of Finance] when needed ?

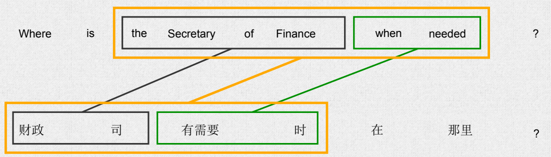
[财政 司] 有需要 时 在 那里 ?

word alignment

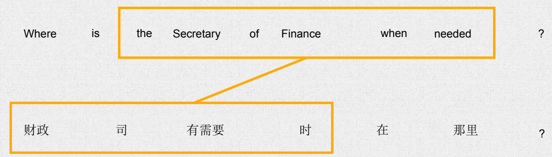
Where is [the Secretary of Finance] [when needed] ?

[财政 司] [有需要 时] 在 那里 ?

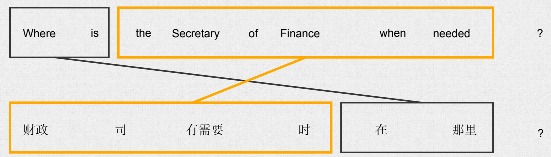
word alignment



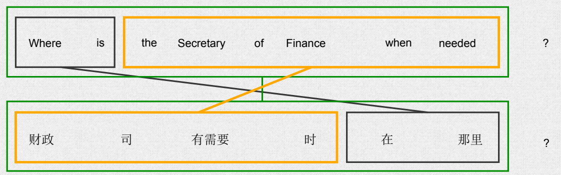
word alignment



word alignment

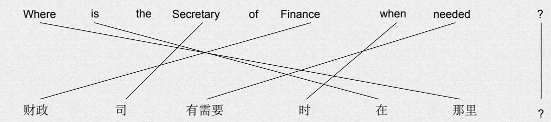


word alignment



word alignment

word alignment



biparsing

aim

- automatically parse an input/output sequence pair

input

- bisequence (input-output pair of sequences)
- token translation lexicon
- language-independent **bracketing ITG** (only 1 generic nonterminal $\mathbf{A}$)

output

- aligned bisequence

Implementation

- biparser

biparsing

1. **Initialization**

$$\delta_{t-1,t,v-1,v}(i) = b_i(\mathbf{e}_t/\mathbf{c}_v), \quad \begin{matrix}1 \le t \le T\\ 1 \le v \le V\end{matrix} \qquad (1)$$

$$\delta_{t-1,t,v,v}(i) = b_i(\mathbf{e}_t/\epsilon), \quad \begin{matrix}1 \le t \le T\\ 0 \le v \le V\end{matrix} \qquad (2)$$

$$\delta_{t,t,v-1,v}(i) = b_i(\epsilon/\mathbf{c}_v), \quad \begin{matrix}0 \le t \le T\\ 1 \le v \le V\end{matrix} \qquad (3)$$

2. **Recursion**

For all i,s,t,u,v such that $\left\{\begin{matrix}1\le i\le N\\ 0\le s<t\le T\\ 0\le u<v\le V\\ t-s+v-u>2\end{matrix}\right.$

$$\delta_{stuv}(i) = \max[\delta^{[]}_{stuv}(i), \delta^{\langle\rangle}_{stuv}(i)] \qquad (4)$$

$$\theta_{stuv}(i) = \begin{cases} [\,] & \text{if } \delta^{[]}_{stuv}(i) \ge \delta^{\langle\rangle}_{stuv}(i)\\ \langle\rangle & \text{otherwise}\end{cases} \qquad (5)$$

where

$$\delta^{[]}_{stuv}(i) = \max_{\substack{1\le j\le N\\ 1\le k\le N\\ s\le S\le t\\ u\le U\le v\\ (S-s)(t-S)+(U-u)(v-U)\ne 0}} a_{i\to[jk]}\, \delta_{sSuU}(j)\, \delta_{StUv}(k) \qquad (6)$$

$$\begin{bmatrix}\iota^{[]}_{stuv}(i)\\ \kappa^{[]}_{stuv}(i)\\ \sigma^{[]}_{stuv}(i)\\ \upsilon^{[]}_{stuv}(i)\end{bmatrix} = \operatorname*{argmax}_{\substack{1\le j\le N\\ 1\le k\le N\\ s\le S\le t\\ u\le U\le v\\ (S-s)(t-S)+(U-u)(v-U)\ne 0}} a_{i\to[jk]}\, \delta_{sSuU}(j)\, \delta_{StUv}(k) \qquad (7)$$

$$\delta^{\langle\rangle}_{stuv}(i) = \max_{\substack{1\le j\le N\\ 1\le k\le N\\ s\le S\le t\\ u\le U\le v\\ (S-s)(t-S)+(U-u)(v-U)\ne 0}} a_{i\to\langle jk\rangle}\, \delta_{sSUv}(j)\, \delta_{StuU}(k) \qquad (8)$$

$$\begin{bmatrix}\iota^{\langle\rangle}_{stuv}(i)\\ \kappa^{\langle\rangle}_{stuv}(i)\\ \sigma^{\langle\rangle}_{stuv}(i)\\ \upsilon^{\langle\rangle}_{stuv}(i)\end{bmatrix} = \operatorname*{argmax}_{\substack{1\le j\le N\\ 1\le k\le N\\ s\le S\le t\\ u\le U\le v\\ (S-s)(t-S)+(U-u)(v-U)\ne 0}} a_{i\to\langle jk\rangle}\, \delta_{sSUv}(j)\, \delta_{StuU}(k) \qquad (9)$$

3. **Reconstruction**

Initialize by setting the root of the parse tree to $q_1 = (0,T,0,V)$ and its nonterminal label to $\ell(q_1) = S$. The remaining descendants in the optimal parse tree are then given recursively for any $q = (s,t,u,v)$ by:

$$\textsc{Left}(q) = \begin{cases} \text{NIL} & \text{if } t-s+v-u\le 2\\ (s, \sigma^{[]}_q(\ell(q)), u, \upsilon^{[]}_q(\ell(q))) & \text{if } \theta_q(\ell(q)) = [\,] \text{ and } t-s+v-u>2\\ (s, \sigma^{\langle\rangle}_q(\ell(q)), \upsilon^{\langle\rangle}_q(\ell(q)), v) & \text{if } \theta_q(\ell(q)) = \langle\rangle \text{ and } t-s+v-u>2\end{cases} \qquad (10)$$

$$\textsc{Right}(q) = \begin{cases} \text{NIL} & \text{if } t-s+v-u\le 2\\ (\sigma^{[]}_q(\ell(q)), t, \upsilon^{[]}_q(\ell(q)), v) & \text{if } \theta_q(\ell(q)) = [\,] \text{ and } t-s+v-u>2\\ (\sigma^{\langle\rangle}_q(\ell(q)), t, u, \upsilon^{\langle\rangle}_q(\ell(q))) & \text{if } \theta_q(\ell(q)) = \langle\rangle \text{ and } t-s+v-u>2\end{cases} \qquad (11)$$

$$\ell(\textsc{Left}(q)) = \iota_q^{\theta_q(\ell(q))}(\ell(q)) \qquad (12)$$

$$\ell(\textsc{Right}(q)) = \kappa_q^{\theta_q(\ell(q))}(\ell(q)) \qquad (13)$$

(Wu, IJCAI 1995)

2-normal form for ITGs

Lemma 1
For any inversion transduction grammar G, there exists an equivalent inversion transduction grammar G' where $T(G) = T(G')$, such that:

1. If $\epsilon \in L_1(G)$ and $\epsilon \in L_2(G)$, then G' contains a single production of the form $S' \to \epsilon/\epsilon$, where S' is the start symbol of G' and does not appear on the right-hand side of any production of G';
2. otherwise G' contains no productions of the form $A \to \epsilon/\epsilon$.

Lemma 2
For any inversion transduction grammar G, there exists an equivalent inversion transduction grammar G' where $T(G) = T(G')$, such that the right-hand side of any production of G' contains either a single terminal-pair or a list of nonterminals.

Lemma 3
For any inversion transduction grammar G, there exists an equivalent inversion transduction grammar G' where $T(G) = T(G')$, such that G' does not contain any productions of the form $A \to B$.

Theorem 1
For any inversion transduction grammar G, there exists an equivalent inversion transduction grammar G' in which every production takes one of the following forms:

$$\begin{array}{lllllllll} S & \to & \epsilon/\epsilon & A & \to & x/\epsilon & A & \to & [B\,C]\\ A & \to & x/y & A & \to & \epsilon/y & A & \to & \langle B\,C\rangle \end{array}$$

(Wu 1997)

translation-driven segmentation

aim
- avoid premature segmentation errors
- optimize chunk boundaries using bilingual/bimodal clues

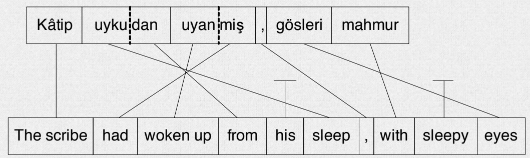
input
- bisequence (input-output pair of sequences)
- token translation lexicon

output
- segmented bisequence

implementation
- segmental biparser, instead of token-based biparser
- integrates segmentation decisions into dynamic programming

(Wu 1997)

translation-driven segmentation



bracketing

aim
- bootstrap grammar induction, with **zero** syntactic knowledge
- use bilingual/bimodal clues to label sequences with tree structure

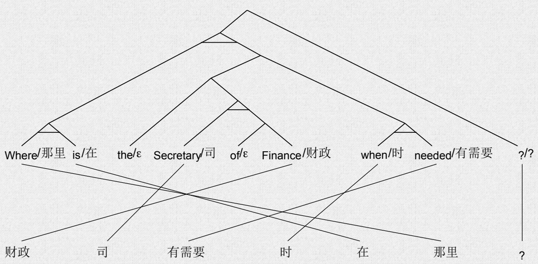
input
- bisequence (input-output pair of sequences)
- token translation lexicon
- language-independent **bracketing ITG** (only 1 generic nonterminal **A**)

output
- bracketing tree structure for **both** sequences

implementation
- biparser (choose either token-based or segmental version)

(Wu, ACL 1995)

bracketing



bracketing

[These/這些 arrangements/安排 will/ε ε/可 enhance/加強 our/我們 ⟨[ε/的 ability/能力] [to/ε ε/日後 maintain/維持 monetary/金融 stability/穩定 in the years to come/ε]⟩ ./。]

[The/ε Authority/管理局 will/將會 ⟨[be/ε accountable/負責] [to the/ε ε/向 Financial/財政 Secretary/司]⟩ ./。]

[They/他們 ⟨ are/ε right/正確 ε/十分 to/ε do/做 ε/這樣 so/ε ⟩ ./。]

[⟨[Even/ε more/更 important/重要] [,/ε however/但]⟩ [,/ε ε/的, is/是 to make the very best of our/ε ε/善用香港 own/本身 ε/的 talent/人才] ./。]

[I/我 hope/ε ε/<>望 employers/僱主 will/會 make full/ε ε/充分善 use/用 [of/ε those/那些] ⟨⟨[ε/的工 who/人] [have acquired/ε ε/學到 new/新 skills/技能]⟩ [through/透過 this/這個 programme/計劃]⟩ ./。]

[I/我 have/已 <> at/ε length/詳細 ⟨ on/ε how/怎樣 we/我們 ε/講述⟩ [can/可以 boost/ε ε/促進 our/本港 ε/的 prosperity/繁榮] ./。]

coercion

aim
- bootstrap grammar induction, knowing only **some other** language
- use an English grammar to parse Chinese
- use bilingual/bimodal clues to coerce Chinese into English tree structure

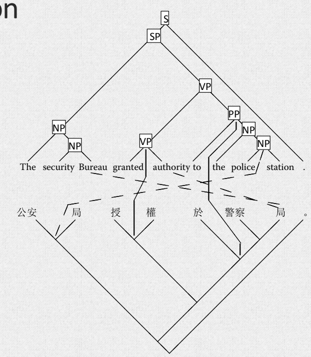
input
- bisequence (input-output pair of sequences)
- token translation lexicon
- input language CFG, with each rule mirrored into straight & inverted ITG rules

output
- labeled tree structure for **output** language sequence

implementation
- biparser (choose either token-based or segmental version)

(Wu, WVLC 1995)

coercion



projection (bilingual constraint transfer)

aim
- bootstrap grammar induction, projecting constraints from another language
- use an English parse to constrain Chinese tree structure
- use bilingual/bimodal clues to coerce Chinese into English tree structure

input
- bisequence, where input sequence has been parsed
- token translation lexicon
- language-independent **bracketing ITG** (only 1 generic nonterminal **A**)

output
- labeled tree structure for **output** language sequence

Implementation
- biparser, incorporating constraints into dynamic programming (hard or soft)

learning phrasal translation lexicons

aim
- bootstrap grammar induction, projecting labels from some other language
- use an English parse to constrain Chinese tree structure
- use bilingual/bimodal clues to coerce Chinese into English tree structure

input
- bisequence
- token translation lexicon
- language-independent **bracketing ITG** (only 1 generic nonterminal **A**)

output
- labeled tree structure for **output** language sequence

Implementation
- biparser, incorporating constraints into dynamic programming (hard or soft)

(Wu, TMI 1995)

learning phrasal translation lexicons

1 % in real	1%的實質
Would you	你是否
an acceptable starting point for this new policy	是可接受為這項新政策的起點
are about 3 . 5 million	大概有350萬
born in Hong	在香港出生
for Hong	為香港
have the right to decide our	有權決定我
in what way the Government would increase their job opportunities ; and	政府如何增加他們的就業機會;及
last month	上個月
never to say " never "	不要說"永不"
reserves and surpluses	儲備和盈餘
starting point for this new policy	為這項新政策的起點
there will be many practical difficulties in terms of implementation	實行時會有很多實際困難
year ended 3 1 March 1 9 9 1	截至一九九一月三十一日

tree alignment

aim
- automatically map constituents between input and output sequences

Input
- bisequence, where input and output sequences have been parsed
- token translation lexicon
- language-independent **bracketing ITG** (only 1 generic nonterminal **A**)

output
- recursively aligned bisequence

Implementation
- biparser

transduction ("decoding")

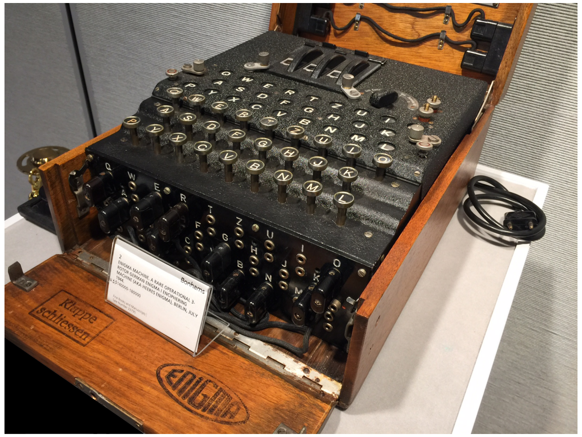
aim
- translate a sequence from an input representation to an output representation, with **zero** syntactic knowledge

Input
- input language sequence
- token translation lexicon
- language-independent **bracketing ITG** (only 1 generic nonterminal **A**)

Output
- output language sequence that is a translation of the input sequence

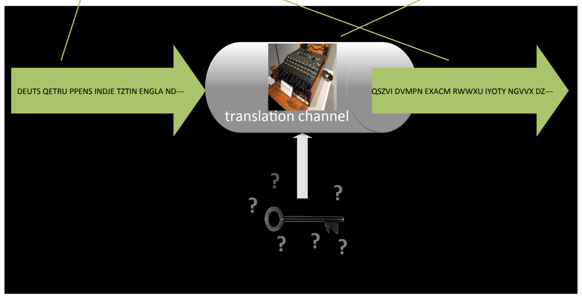
Implementation
- CKY style BITG transducer with output language n-gram model

(Wu, ACL 1996)



optimal German decrypt — encrypted German sentence — Enigma model (translation channel) — language model

$$e^* = \arg\max_e P(e|c) = \arg\max_e \frac{P(c|e)P(e)}{P(c)} = \arg\max_e P(c|e)P(e)$$



an effective language is **elegant**

an effective representation is **elegant**

Turing machines are
too inefficient to learn

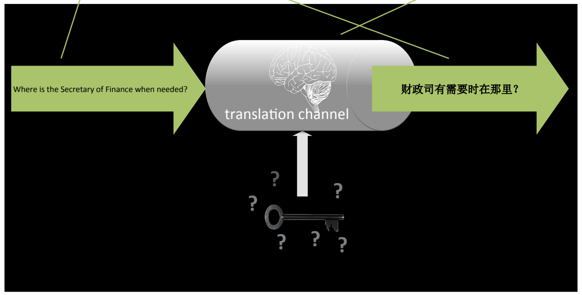
optimal English translation

Chinese input sentence

translation channel

language model

$$e^* = \arg\max_e P(e \mid c) = \arg\max_e \frac{P(c \mid e)P(e)}{P(c)} = \arg\max_e P(c \mid e)P(e)$$



transduction (“decoding”)

aim

translate a sequence from an input representation to an output representation, with **zero** syntactic knowledge

input

- input language sequence
- token translation lexicon
- language-independent **bracketing ITG** (only 1 generic nonterminal **A**)

output

- output language sequence that is a translation of the input sequence

implementation

- CKY style BITG transducer with output language n-gram model

(Wu, ACL 1996)

transduction (“grammatical channel”)

aim

translate a sequence from an input representation to an output representation, knowing only the **output** language syntax

input

- input language sequence
- token translation lexicon
- output language CFG, with each rule mirrored into straight & inverted ITG rules

output

- output language sequence that is a translation of the input sequence

implementation

- Earley style ITG transducer with output language n-gram model

(Wu & Wong, Coling-ACL 1998)

we’ve barely scratched the surface of the many applications of stochastic transduction grammars

- **grammar induction**: automatically decomposing both sides of a parallel corpus (“bilingual bracketing”)
- **biparsing**: bilingual parsing of both sides of a parallel corpus
- **tree alignment** under various constraints to extract or induce transduction rules (patterns, transfer rules, examples, templates)
- **segmentation**: translation-driven word segmentation (“phrase-based”)
- **coercion** or **projection**: parsing one language using the grammar of another language
- **Earley-style biparsing & decoding**: particularly useful for linguistic grammars (which are often not binarized)
- **training**: polynomial-time EM parameter estimation
- **translation** or **decoding**: polynomial-time statistical translation
- **grammatical decoding**: even faster, grammatical statistical machine translation (incorporating linguistic grammars)
- **transduction grammar induction**: fully unsupervised learning of relations

Efficient polynomial-time algorithms for FSTG, LTG, and ITG learning of structural relationships are possible because of the evolutionary drive toward restricted expressiveness of practical languages and concepts in common use

transduction grammar induction

aim

go from a parallel corpus to a transduction grammar

which can then be used as a transducer (translation system)

input

- bicorpus (large set of bisequences)

output

- transduction grammar (typically an ITG)

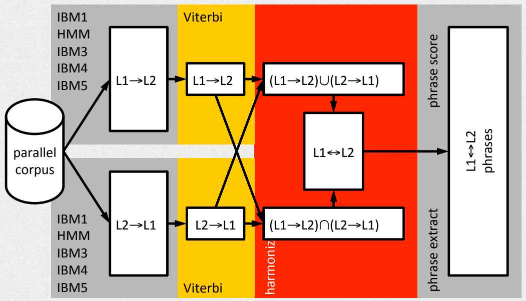
implementation

- incrementally learn TGs from simple to complex
- gradually climb up the complexity hierarchy for transductions

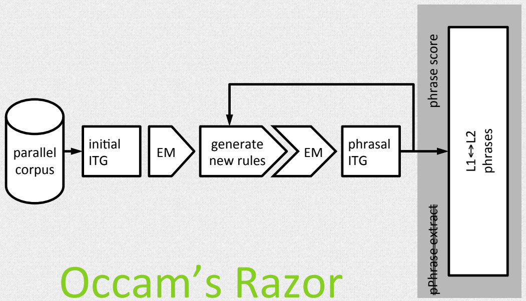
but a lot of SMT today

- relies excessively on
  - memorizing long lexical translations
    - … or heavily lexicalized long transduction rules
- rather than
  - generalizing abstract patterns
  - reusing short lexical translations

compare this…

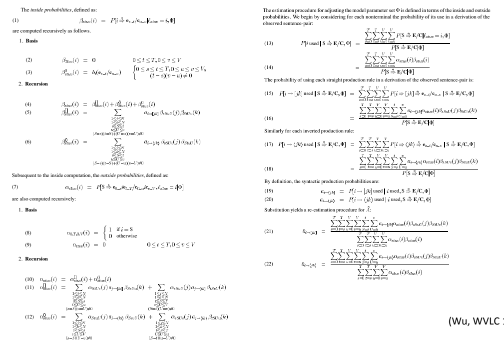


… with this



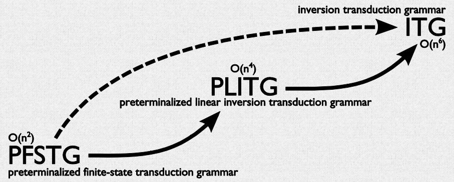
Occam’s Razor

EM algorithm for ITGs



(Wu, WVLC 1995)

bootstrapping



(Saers, Addanki & Wu, Coling 2012)

wait...

you wanna translate with finite-state transducers?!

- no!
  - FSTs are useless for anything but the most simple, monotone translation tasks
- but...
  - complex translation tasks are made up of simple, monotone translation tasks

review

a **grammar** generates a **language**
... which is a set of *sentences*

a **transduction grammar** generates a **transduction**
... which is a set of *sentence pairs*

SEGMENTAL

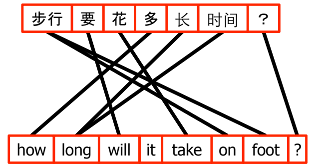
Monolingual Languages / Bilingual Transductions

Monolingual Languages			Bilingual Transductions
regular or finite-state languages **FSA** *or* *CFG that is right or left linear or regular*	$O(n^2)$	$O(n^4)$	regular or finite-state transductions **FST** *or* *SDTG that is right or left linear or regular*
linear languages **LG** *or* *CFG that is linear or unary*	$O(n^2)$	$O(n^4)$	linear transductions **LTG** *or* *SDTG that is linear or unary*
context-free languages **CFG**	$O(n^3)$	$O(n^6)$	inversion transductions **ITG** *or* *SDTG that is binary or ternary or inverting*
		$O(n^{2n+2})$	syntax-directed transductions **SDTG** *(or synchronous CFG)*

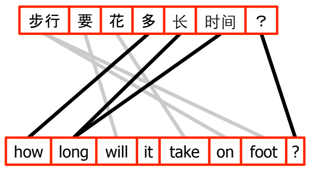
finite-state

- finite-state grammar
  - $S \rightarrow A$
    $A \rightarrow \varepsilon$
    $A \rightarrow e\ B$
- finite-state transduction grammar
  - $S \rightarrow A$
    $A \rightarrow \varepsilon/\varepsilon$
    $A \rightarrow e/f\ B$ substitution
    $A \rightarrow e/\varepsilon\ B$ insertion
    $A \rightarrow \varepsilon/f\ B$ deletion

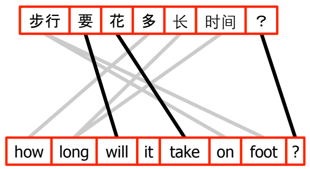
example sentence



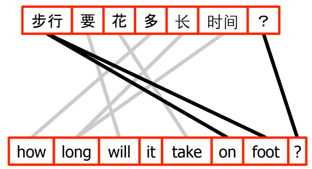
FSTG alignments



FSTG alignments



FSTG alignments



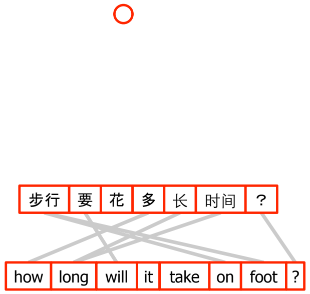
Linear

- Linear grammar
  - S → A
    A → ε
    A → e B
    A → B e
- Linear transduction grammar
  - S → A ; A → [e/f B] ; A → [ε/f B] ; A → ‹ε/f B›
    A → ε/ε ; A → ‹e/f B› ; A → [e/ε B] ; A → ‹e/ε B›
    ~~A → e/f B~~ ; A → [B e/f] ; A → [B ε/f] ; A → ‹B ε/f›
    ~~A → B e/f~~ ; A → ‹B e/f› ; A → [B e/ε] ; A → ‹B e/ε›

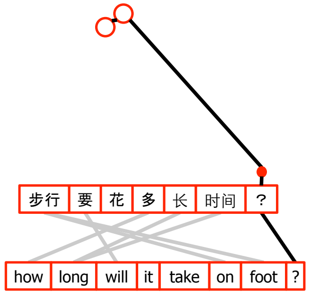
Lots of rule forms…

- … for the same token pairs (biterminals)
- Risks diluting the statistics
- Introduce preterminalized transduction grammars
- Linear transduction grammar
  - S → A ; S → A
    A → ε/ε ; A → ε/ε
    A → [e/f B] ; A → [P B], ~~P → e/f~~ ; P → e/f
    A → ‹e/f B› ; A → ‹P B›, ~~P → e/f~~ ; P → e/ε
    A → [B e/f] ; A → [B P], ~~P → e/f~~ ; P → ε/f
    A → ‹B e/f› ; A → ‹B P›, ~~P → e/f~~

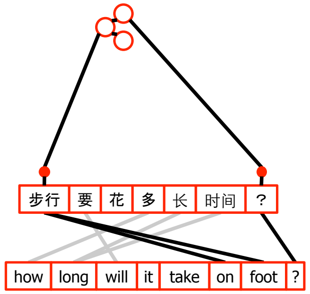
PLITG parsing



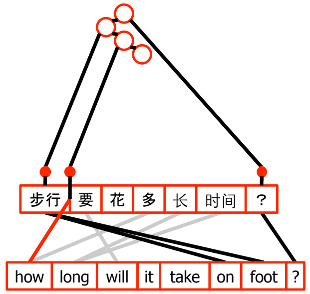
PLITG parsing



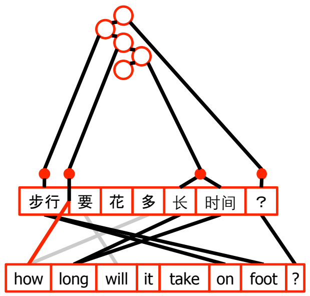
PLITG parsing



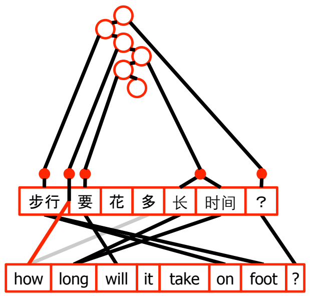
PLITG parsing



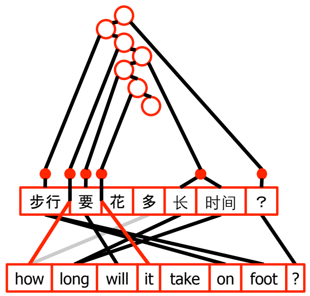
PLITG parsing



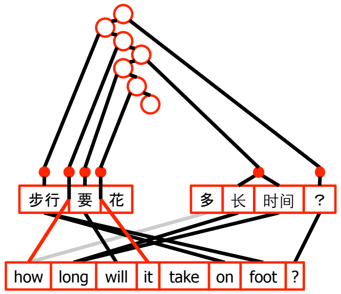
PLITG parsing



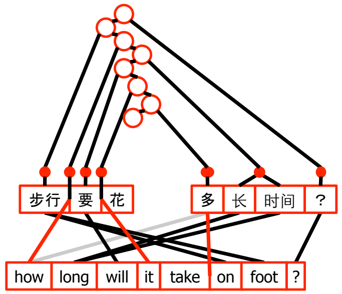
PLITG parsing



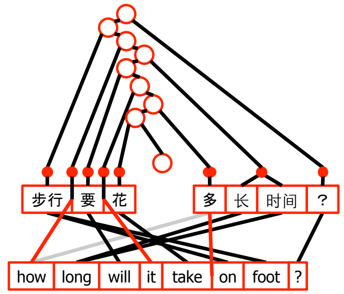
PLITG parsing



PLITG parsing



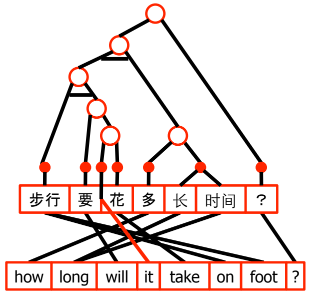
PLITG parsing



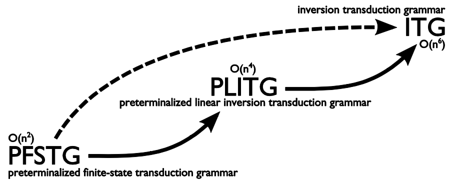
ITG

- context-free grammar (2-normal form)
  - S → A
    A → B C
    A → e
- inversion transduction grammar
  - S → A
    A → [B C]
    A → ‹B C›
    A → e/f
    A → e/ε
    A → ε/f

ITG parsing



Roadmap



Grammar conversion: PFSTG-PLITG

PFSTG	PLITG
S → A	S → A
A → ε/ε	A → ε/ε
A → P B	A → [P B]
	A → ‹P B›
	A → [B P]
	A → ‹B P›
P → e/f	P → e/f

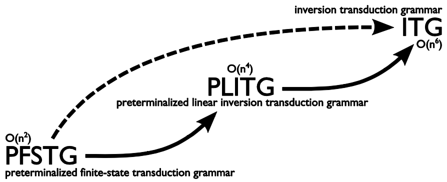
Grammar conversion: PLITG-ITG

- Not as straight forward
- Idea: "promote" preterminals to be proper nonterminals
  - P → e/f becomes P → e/f and P → A
  - p'(e/f|P) = α p(e/f|P),
    p'(A|P) = (1 - α) β(A|P) p(e/f|P)
    - α = 0.5,
      β(A|P) = uniform over nonterminals
  - Perform standard grammar normalization to eliminate nullary and unary rules

Grammar conversion: PFSTG-ITG

- Cheat!
- Compose the previous conversions
  PFSTG-PLITG • PLITG-ITG
  gives
  PFSTG-ITG
- No training at the PLITG stage

roadmap



Fun with simple grammars

- Splitting
  - Split one nonterminal or preterminal symbol into n new symbols
  - Apply controlled perturbation to split the probability mass
  - (See the paper for details)

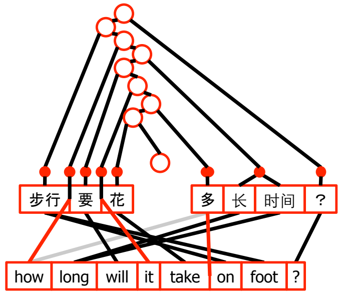
Splitting helps!

- Assume two rules:
  - A → [P A]
  - P → will/要
- We want to:
  - Split A into A and B
  - Split P into P and Q
- Resulting rules:
  - A → [P A], A → [P B], A → [Q A], A → [Q B]
    B → [P A], B → [P B], B → [Q A], B → [Q B]
  - P → will/要, Q → will/要

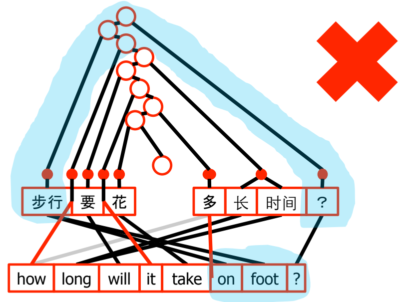
Fun with simple grammars

- Chunking
  - Any sequence of two contiguous terminal productions found in the parse forest could be one production
  - Each round of chunking doubles the potential phrase length
  - Time consuming on large forests
  - (See our EAMT and Interspeech paper from last year for details)

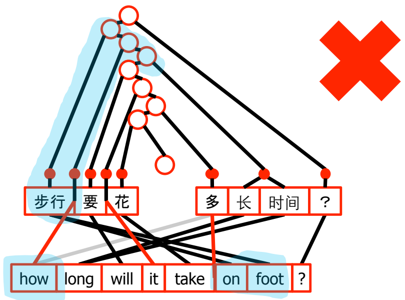
Chunking helps!



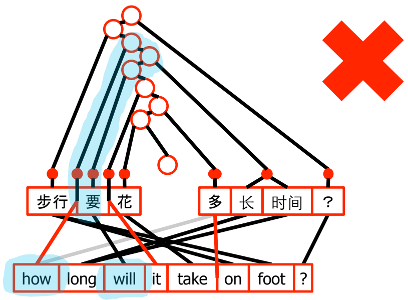
Chunking helps!



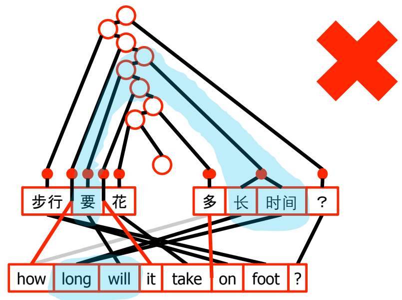
Chunking helps!



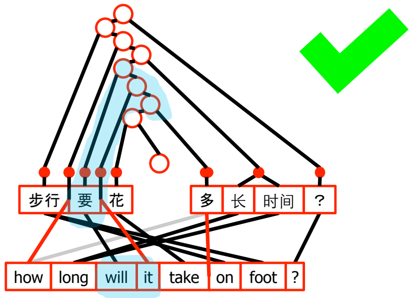
Chunking helps!



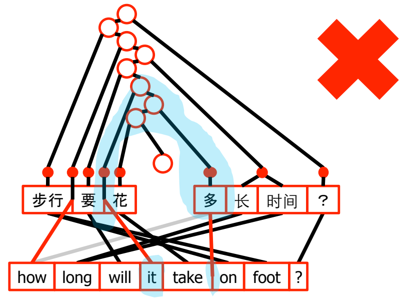
Chunking helps!



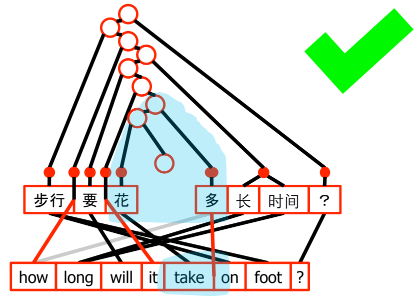
Chunking helps!



Chunking helps!



Chunking helps!



experimental setup

- Initialize a PFSTG from a parallel corpus
  - IWSLT07 Chinese-English
- Chunk and split to your heart's desire
- Move to PLITG then to ITG
  or move straight to ITG
  - Traverse the roadmap
- Train at each stage
- Measure cross-entropy

why is cross-entropy on the training set important?

- Low cross-entropy may be indicative of over-fitting
- Over-fitting to a lower cross-entropy is indicative of model fit
- PFSTGs are unable to over-fit to the same level that ITGs are able to
- Indicates that ITGs as a model is a better fit to the problem

lessons
ways to lower the entropy

- **Baseline:** cross-entropy 110.2 (PFSTG)
- **Best:** cross-entropy 32.8
  PFSTG-chunk-chunk-split-split-split-ITG
  - Nice: still fits in 16Gb RAM
- **Promising candidate:** cross-entropy 35.9
  PFSTG-chunk-chunk-chunk-ITG
  - After only 1 chunk: 60.7
  - After chunk-chunk: 36.5 (already mostly there!)
  - Unfortunately, couldn't split after chunk-chunk-chunk within 16Gb RAM

lessons
ways to lower the entropy

- In general
  - ITG is better than PFSTG and PLITG
  - Chunking helps
  - Splitting helps

remember what we want

- big parallel corpus → small transduction grammar
- learn lexical phrase translations
  (i.e. a segmental transduction grammar)
- compact generalization of the translation knowledge encoded in the corpus
- unsupervised learning of transduction grammar rules without Giza, Moses, parsers, or anything else

why?

- rearchitecting the SMT core: "Machine Learning 101"
  - Do training and testing on the **same** model
  - Get the **inductive bias** right: core internal representation designed from the start for learning semantic frame generalizations
  - Emphasis on **generalizing** rather than memorizing
  - Minimum description length / MAP → **Occam's razor** for model size
- evaluated in pure, unadulterated form
  - Not as a preprocessing subroutine (eg, for word alignment) within an off-the-shelf "stack-of-hacks" SMT spaghetti architecure
  - ITG decoder matched to ITG learner
  - We'd rather see lower BLEU scores temporarily, so we can better understand transduction grammar induction behavior
  - Don't obscure your model by burying it within a big "stack-of-hacks"!

common SMT training pipeline

parallel corpus → word align → sym-metrize → extract phrases → score phrases → translation model

- Long pipeline propagates errors: risk of premature commitment
  - no way to recover from mistakes in earlier steps
- No credit/blame assignment during learning!
- To try to compensate:
  - massively over-generate phrases
  - result: heavy bias toward memorizing huge corpus instead of learning the right abstract generalizations
- Same pipeline problem for Hierarchical/Syntactic {tree, string} to {tree, string}
  - parse parallel corpus where the trees go
  - replace "phrases" with TG rules

our training "pipeline"

parallel corpus → ITG learning algorithm → translation model

- No pipeline
- No risk of premature commitment
- Replaces many intermediate learning steps
  - … which, admittedly, have been engineered with heuristic tweaks over a long time
  - worth it to carefully understand correct unsupervised learning of generalizations

our specific SSBITG model

- Unsupervised induction of stochastic segmental bracketing inversion transduction grammars

lacks nonterminal categories

has probabilities

phrasal translation lexicon

Our specific SSBITG model

- Unsupervised induction of stochastic segmental bracketing inversion transduction grammars
  - Compact generalization of the translation knowledge encoded in the corpus

Our specific SSBITG model

- Unsupervised induction of stochastic segmental bracketing inversion transduction grammars
  - Compact generalization of the translation knowledge encoded in the corpus
- Bayesian learning objective
  - Closely related to minimum description length
  - Structural prior based on minimum description length
  - Dirichlet parameter prior
  - Fixed model type prior

Our specific SSBITG model

- Unsupervised induction of stochastic segmental bracketing inversion transduction grammars
  - Compact generalization of the translation knowledge encoded in the corpus
- Bayesian learning objective
  - Closely related to minimum description length
  - Structural prior based on minimum description length
  - Dirichlet parameter prior
  - Fixed model type prior
- Structural search based on iteratively segmenting sentence pairs

How to induce transduction rules?

- Can we beat our COLING 2012 bottom-up chunking method?
- Test by using induced transduction grammar directly to translate unseen sentences

Opposite search strategy "directions" for learning transduction rules

- Chunk rules bottom-up (COLING 2012)
  - Start with very small lexical equivalences (biterminals)
  - Look for promising chunks during biparsing
  - Make the chunks explicit biterminals
  - Repeat
- Segment rules top-down (IJCNLP 2013)
  - Start with all sentence pairs as biterminals
  - Segment the existing biterminals into smaller chunks
  - Make the smaller chunks explicit biterminals
  - Repeat

Bottom-up rule chunking strategy

1. initialize a token-based FSTG
2. parse the bicorpus
3. assume that any two adjacent lexical productions could have been generated with a single **chunked** rule, and add them
4. train the FSTG using EM
5. transform the FSTG into an LITG and train using EM
6. transform the LITG into an ITG and train using EM

Saers, Markus, Karteek Addanki & Dekai Wu (2012) "From Finite-State to Inversion Transductions: Toward Unsupervised Bilingual Grammar Induction" in *COLING 2012*

Opposite search strategy "directions" for learning transduction rules

- Chunk rules bottom-up (COLING 2012)
  - Start with very small lexical equivalences (biterminals)
  - Look for promising chunks during biparsing
  - Make the chunks explicit biterminals
  - Repeat
- Segment rules top-down (IJCNLP 2013)
  - Start with all sentence pairs as biterminals
  - Segment the existing biterminals into smaller chunks
  - Make the smaller chunks explicit biterminals
  - Repeat

Opposite search strategy "directions" for learning transduction rules

- Chunk rules bottom-up (COLING 2012)
  - Start with very small lexical equivalences (biterminals)
  - Look for promising chunks during biparsing
  - Make the chunks explicit biterminals
  - Repeat
- Segment rules top-down (IJCNLP 2013)
  - Start with all sentence pairs as biterminals
  - Segment the existing biterminals into smaller chunks
  - Make the smaller chunks explicit biterminals
  - Repeat

Segmentation, intuitively

- five thousand yen is my limit
  我最多出五千日元
- the total fare is five thousand yen
  总共的费用是五千日元

Segmentation, intuitively

- is my limit
  我最多出
- the total fare is
  总共的费用是
- five thousand yen
  五千日元

SHORTER

New search strategy
Top-down rule-segmenting strategy

1. **initialize** an ITG containing each sentence pair as a biterminal
2. **foreach** "frequently shared biaffix"
3. **hypothesize** the set of segmentations that it suggests
4. **evaluate** the delta in ITG "goodness" the set would cause*
5. **commit** greedily to good hypothesis sets
6. **goto** 2

* for present purposes, "goodness" = description length

Iterative rule segmentation
as phrasal translation lexicon search

Q. How to suggest possible ways to segment rules?

A. Look for frequently shared biaffixes

- Each biaffix suggests a set of rule segmentation hypotheses
  - 4 types of rule segmentations can be suggested

- Can be efficiently computed
- Estimate delta in objective function for each

"Goodness" of an ITG
If objective function is description length...

- #bits needed to encode the model $\mathrm{DL}(\Phi)$
  - Φ = model = ITG

plus...

- #bits needed for the model to encode the data $\mathrm{DL}(D|\Phi)$
  - D = data = parallel training corpus

Top-down rule segmentation

31 symbols

S → A
A → five thousand yen is my limit/我最多出五千日元

A → the total fare is five thousand yen/总共的费用是五千日元

Top-down rule segmentation

31 symbols

S → A
A → five thousand yen is my limit/我最多出五千日元
A → 〈AA〉
A → five thousand yen/五千日元
A → is my limit/我最多出
A → the total fare is five thousand yen/总共的费用是五千日元
A → [AA]
A → the total fare is/总共的费用是
A → five thousand yen/五千日元

Top-down rule segmentation

24 symbols

S → A

A → ⟨AA⟩
A → five thousand yen/五千日元
A → is my limit/我最多出

A → [AA]
A → the total fare is/总共的费用是
A → five thousand yen/五千日元

Description length of an ITG

- Serialize grammar into a message
- Measure number of bits needed to encode the message
- Example:

$$S \to A \qquad A \to \langle AA \rangle \qquad A \to [AA]$$
$$A \to \text{have}/\text{有} \qquad A \to \text{yes}/\text{有} \qquad A \to \text{yes}/\text{是}$$

- Becomes the message:
  []SA⟨⟩AAA[]AAA[]Ahave有[]Ayes有[]Ayes是
- Assume each symbol requires –lg 1/*N* bits (where *N* is the total number of symbols)
- The above message contains 8 unique symbols → 3 bits each
- The message is 23 symbols long, and needs (23 · 3 =) 69 bits to encode

Minimum Description Length objective

- Want: $\underset{\Phi}{\operatorname{argmin}} DL(\Phi, D)$
- $DL(\Phi, D) \propto DL(D \mid \Phi) + DL(\Phi)$
- $DL(D \mid \Phi) = -\lg P(D \mid \Phi)$
- $DL(\Phi) \approx$ #bits needed to encode model

MAP vs. MDL

- Maximum *a posteriori* probability
- Minimum description length
- Relationship:
  - DL(x) = –lg P(x)
  - P(x) = $2^{-DL(x)}$
- Enables:
  - probabilistic formulation of description length search
  - description length formulation of probabilistic search

Bayesian search (MAP)

- Maximize *a posteriori* probability of the model given the parallel corpus:

$$P(\Phi|D) = \frac{P(\Phi)\, P(D|\Phi)}{P(D)}$$

- Decompose the model prior such that:

$$P(\Phi) = P(\Phi_G)\, P(\Phi_S|\Phi_G)\, P(\theta_\Phi|\Phi_S, \Phi_G)$$

$P(\Phi_G)$ = bracketing inversion transduction grammar

$$P(\Phi_S|\Phi_G) = 2^{-\mathrm{DL}(\Phi_S|\Phi_G)}$$

$P(\theta_\Phi|\Phi_S, \Phi_G)$ = symmetric Dirichlet distribution (α=2)

Bayesian search (MAP)

- Full search problem:

$$\underset{\Phi_G,\Phi_S,\theta_\Phi}{\operatorname{argmax}} P(\Phi_G) \times P(\Phi_S|\Phi_G) \times P(\theta_\Phi|\Phi_S, \Phi_G) \times P(D|\Phi_G, \Phi_S, \theta_\Phi)$$

- In MDL:

$$\underset{\Phi_G,\Phi_S,\theta_\Phi}{\operatorname{argmin}} \mathrm{DL}(\Phi_G) + \mathrm{DL}(\Phi_S|\Phi_G) + \mathrm{DL}(\theta_\Phi|\Phi_S, \Phi_G) + \mathrm{DL}(D|\theta_\Phi, \Phi_S, \Phi_G)$$

evaluating the delta in P(D|Φ)

- Requires biparsing for every hypothesized new Φ
- Intractable
- … must approximate

estimating the delta in P(D|Φ)

- Assume that r_0 is segmented into r_1, r_2 and r_3
- Approximation assumption:

$$\frac{P(D|\Phi'_S, \Phi_G, \theta_{\Phi'})}{P(D|\Phi_S, \Phi_G, \theta_\Phi)} = \frac{\hat{p}'(r_1)\, \hat{p}'(r_2)\, \hat{p}'(r_3)}{\hat{p}(r_0)}$$

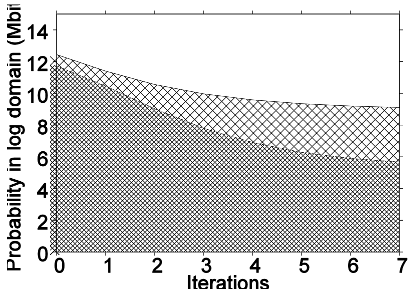
- The new rule probability function $\hat{p}'$ will be:

$$\begin{aligned}
\hat{p}'(r_0) &= 0 \\
\hat{p}'(r_1) &= \hat{p}(r_1) + \tfrac{1}{3}\hat{p}(r_0) \\
\hat{p}'(r_2) &= \hat{p}(r_2) + \tfrac{1}{3}\hat{p}(r_0) \\
\hat{p}'(r_3) &= \hat{p}(r_3) + \tfrac{1}{3}\hat{p}(r_0)
\end{aligned}$$

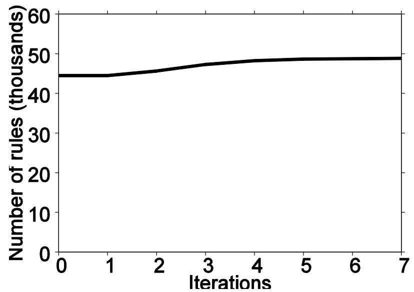
transduction grammar induction by top-down segmenting transduction rules

```
G                      // The grammar
biaffixes_to_rules     // Maps biaffixes to the rules they occur in
biaffixes_delta = []   // Hypothesized biaffixes' impact on P(D|G)
for each biaffix b :
  delta = eval_dl(b, biaffixes_to_rules[b], G)
  if (delta < 0)
    biaffixes_delta.push(b, delta)
sort_by_delta(biaffixes_delta)
for each b:delta pair in biaffixes_delta :
  real_delta = eval_dl(b, biaffixes_to_rules[b], G)
  if (real_delta < 0)
    G = make_segmentations(b, biaffixes_to_rules[b], G)
```

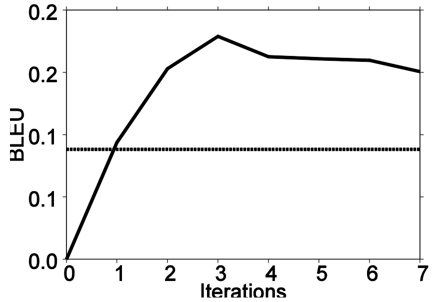
impact of model structure
a posteriori probability during learning (log domain)



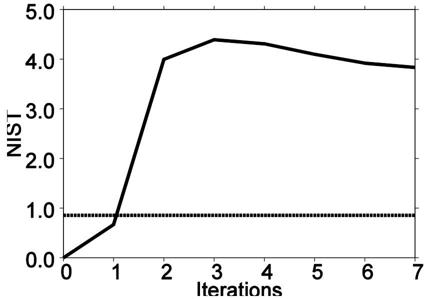
Impact of model structure
rule count during learning (log domain)



BLEU score



NIST score



(in table form)

System	NIST	BLEU
baseline	0.8554	8.83
initial	0.0000	0.00
iteration 1	0.6686	9.38
iteration 2	3.9976	15.30
iteration 3	**4.3928**	**17.89**
iteration 4	4.3122	16.26
iteration 5	4.0981	16.10
iteration 6	3.9191	15.97
iteration 7	3.8338	15.06

MDL/MAP transduction grammar induction

- Bayesian MAP quite promising for driving ITG induction via top-down segmentation of rules
  - closely related to MDL (DL is more natural for ITG structure prior)
- Beats bilingual lexical chunking driven by ML
  - learns a much smaller ITG…
  - … that performs better on held-out test data
- **New!** even better results combining chunking and segmentation
- Rearchitecting the SMT core: "Machine Learning 101"
  - **inductive bias** – internal representation is set up from the start for learning semantic frame generalizations
  - **learns small models** – less reliance on memorizing huge corpora
  - **rapidly improving** – newer results already into mid-20s BLEU range without Giza, Moses, parsers, or anything else
  - **representational transparency** – error analyses to understand learning properties

escaping from blocks world

Can the same exact **core capability** be used not only for conventional AI tasks, but also for all sorts of creative tasks?

Prof Dekai Wu HKUST Language, music, and creativity Transduction grammar induction & applications

transduction grammars simultaneously model Boden's (1992) three types of creativity

combinational new combinations of familiar ideas

exploratory generation of new ideas by exploration of a space of concepts

transformational involves a transformation of the search space so new kinds of ideas can be generated

what

makes

music

music

?

the many languages of music

lyrics	sequences of words
melodies	sequences of notes
chord progressions	sequences of chords
rhythms	sequences of percussive hits
meters	sequences of pulses
ostinatos	sequences of repetitions
stanzas	sequences of lines
verses	sequences of stanzas
songs	sequences of verses
dynamics	sequences of volumes

the many languages of music

lyrics	sequences of words
melodies	sequences of notes
chord progressions	sequences of chords
rhy…	…e hits
m…	sequences of pulses
osti…	…ns
sta…	
verses	sequences of stanzas
songs	sequences of verses
dynamics	sequences of volumes

if you master one of these languages, are you a musician?

it's about the relationships

- the relationships between multiple different musical languages differentiate aesthetically pleasing music from jarring noise
- internalizing these relationships accounts for
- expectation
- surprise
- resolution

learning the relationships between different languages

technically: learning the <u>transduction</u> that relates two languages

Why apply NLP to music?

- Music
  - is a form of language
  - has had major impact across all human cultures
  - emerges from similar cognitive process as speech and written language (McMullen and Saffran 2004)
- Applying NLP to music
  - similar generalizations to be captured in music and natural language
  - distinguishes well-motivated learning methods from language specific fixes
  - adaptation of statistical NLP models presents interesting challenges

Hip hop is spoken language.

- **rap** is one of the world's most popular forms of spoken language (for decades!) – arguably spoken language's most significant development in ages
- yet inexplicably ignored in language research
- **musical lyrics** way more challenging than classical poetry due to absence of traditional constraints
  - far fewer meter restrictions
  - variable rhyme schemes
  - unusual vocabulary (*Bill Gates ≈ 40 ill dates*)
- user generated content available online
- but off-the-shelf NLP tools not suitably designed

freestyling

freestyling

- **freestyle**: improvisational style of rap performed "off the top of the head" with no previously composed lyrics
- **freestyle battle**: contest of rappers dueling by challenging and responding using improvised lyrics

freestyling

- **freestyle**: improvisational style of rap performed "off the top of the head" with no previously composed lyrics
- **freestyle battle**: contest of rappers dueling by challenging and responding using improvised lyrics

Steve Jobs vs. Bill Gates (source: Epic Rap Battles)

Jobs A man uses the machines you built to sit down and pay his taxes
A man uses the machines I built to listen to the Beatles while he relaxes

Gates Well Steve, you steal all the credit for work that other people do
Did your fat beard Wozniak write these raps for you too?

verse vs. stanza vs. line

All right stop collaborate and listen
Ice is back with my brand new invention
Something grabs a hold of me tightly
Flow like a harpoon daily and nightly
Will it ever stop yo, I don't know
Turn off the lights, and I'll glow
To the extreme I rock a mic like a vandal
Light up a stage and wax a chump like a candle
Dance go rush to the speaker that booms
I'm killing your brain like a poisonous mushroom
Deadly when I play a dope melody
Anything less than the best is a felony
Love it or leave it you better gain weight
You better hit bull's eye the kid don't play
If there was a problem, yo I'll solve it
Check out the hook while my DJ revolves it

verse

verse vs. stanza vs. line

All right stop collaborate and listen
Ice is back with my brand new invention
stanza AA
Something grabs a hold of me tightly
Flow like a harpoon daily and nightly
Will it ever stop yo, I don't know
Turn off the lights, and I'll glow
stanza AABA
To the extreme I rock a mic like a vandal
Light up a stage and wax a chump like a candle
Dance go rush to the speaker that booms
I'm killing your brain like a poisonous mushroom
Deadly when I play a dope melody
Anything less than the best is a felony
Love it or leave it you better gain weight
You better hit bull's eye the kid don't play
If there was a problem, yo I'll solve it
Check out the hook while my DJ revolves it

verse

verse vs. stanza vs. line

line
All right stop collaborate and listen
Ice is back with my brand new invention
stanza AA
Something grabs a hold of me tightly
Flow like a harpoon daily and nightly
Will it ever stop yo, I don't know
Turn off the lights, and I'll glow
stanza AABA
To the extreme I rock a mic like a vandal
Light up a stage and wax a chump like a candle
Dance go rush to the speaker that booms
I'm killing your brain like a poisonous mushroom
Deadly when I play a dope melody
Anything less than the best is a felony
Love it or leave it you better gain weight
You better hit bull's eye the kid don't play
If there was a problem, yo I'll solve it
Check out the hook while my DJ revolves it

verse

freestyling as transduction

(Wu, Addanki, Beloucif, Saers; EMNLP 2013, Interspeech 2013, LREC 2014, IJCAI 2015, ICMC 2015)

FREESTYLE model
hip hop challenge-response as MT

- **approach** – learn to "translate" any line of hip hop to produce improvised, rhyming lyrics
- **requirements** – the underlying model must have
  - strong enough inductive bias for learning to generate responses even without any linguistic or phonetic knowledge
  - sufficient expressive capacity to represent structural relationship between lyrical lines
- **model** – bracketing inversion transduction grammar (BITG) induction
  - rule learning via chunking vs segmentation
  - data selection via adjacent lines vs rhyme scheme detection
  - for English vs Maghrebi French hip hop

freestyling by learning transduction grammars

FREESTYLE model
learning to transduce challenges into responses

- **transduction** – a relation between languages
- **transduction grammar** – ideal for representing structural relationships between lyrical lines
  - model associations between words/chunks to generate fluent and rhyming responses
- **transduction grammar induction** – learn relation between challenges and responses
  - good inductive bias for generating lyrics without any prior linguistic or phonetic knowledge
- **transduce** – use induced transduction grammar to respond to any challenge rap with an improvised, rhyming response

experiments

transducing challenges to responses

- HKUST ITG decoder
  - bottom-up CKY style parsing algorithm with cube pruning (Chiang 2008)
- monotonic constraint
  - bias to match the rhyming order of challenge

transducing challenges to responses

- HKUST ITG decoder
  - Earley style transducer supports arbitrary non-normal-form, non-binarized ITGs of any rank with mixed terminals/nonterminals in rules (Wu & Wong 1998)
  - segmental ITGs support any length lexical phrase translations (Wu 1997)
  - cube pruning (Chiang 2008)
- penalize reflexive rules like A → *yeah* / *yeah*
  - challenge and response identical in ~5% of training instances (chorus lines)
  - danger: reflexive rules that map a line back to itself get induced
  - so decoder penalizes responses too similar to the challenges
  - also decoder can penalize inverted rules as a bias to match rhyming order

challenge	response
hello hello hello hello hello	*hello hello hello hello hello*
yeah yeah yeah yeah yeah	*yeah yeah yeah yeah yeah*
they call me superman…	*they call me superman…*
now can i get it yo	*now can i get it yo*

issues for evaluation

- systems to be judged on quality of responses
- but what is "quality of responses"?
  - absence of improvised "references"
  - extremely subjective akin to "translation"
  - no automatic metrics like BLEU exist
  - multiple evaluation criterion
  - larger context necessary for evaluation

evaluation criterion

- human evaluators
  - native English speakers
  - frequent hip hop listeners
- humans asked to evaluate system responses on
  - **rhyming** – degree of rhyming with challenge
  - **fluency** – "sing"-ability of the challenge-response pair
- each system response was rated *good*, *acceptable* or *bad* for both criterion

the good, the bad, and the acceptable

challenge	response	fluency	rhyming
cause you and your friends ain't nothing but fiends	you know that you me can you a dream	*acceptable*	*bad*
cause you ain't going home till the early **morn**	and the you this alone i i gotta **on**	*bad*	*acceptable*
created different elements and they **travelled** abroad	so you rather to the **well** land	*good*	*acceptable*
man i die to see em all thun i just don't **care**	in the sky and me the in polla and the you **there**	*acceptable*	*good*
what would i **do**	just me and **you**	*good*	*good*
almost a thought	what is this	*good*	*bad*
we fell off into a club to try to **pop** it to daz	you was a of the love i like a **stop** the the i	*bad*	*good*

corpus & contrastive baseline

- data
  - 52,000 lyrics, 260,000 verses
  - 4.2M tokens and 153,000 token types
  - Small fraction of Arabic, French and Spanish lyrics
- phrase based SMT baseline (PBSMT)
  - evaluate out-of-the-box SMT performance
  - standard Moses baseline
  - 4-gram LM trained on all the lyrics

data selection?

disfluency in hip hop lyrics

- ~10% of data had successive repetitions of words like *the* and *I*
- disfluencies typically result from repetitive chants, exclamations, and interjections in lyrics

chorus style lyrics	"hypeman" style backing vocal lyrics
i i i I *oh oh oh oh ahhh* *rock rock rock the boat* *i i i i i ice-t ice-t* *yes yes yes a yes yes y'all*	*hey hey like like like that like that*

- compare two disfluency handling strategies
  - **filtering** – remove lines with disfluencies
  - **correction** – replace all successive repetitions (*the the the* → *the*)

creating training data
how do we select challenge-response pairs?

- need – lots of training examples consisting of
  - a line of rap, with
  - a fluent and salient rhyming response
- naïve approach – all pairs of lines in the same stanza
  - explodes the training data size
  - does not capture rhyming dependencies
- better approach 1 – all successive line pairs in the same stanza
  - keeps training set size proportional to rap corpus size
  - but still does not ensure that training examples rhyme
- better approach 2 ("**RS**") – only successive line pairs that rhyme
  - but how can we know which line pairs actually rhyme?

rhyme scheme detection

rhyme scheme detection
(Addanki & Wu, SLSP 2013)

- in keeping with our linguistics-lite model – we don't use a pronunciation dictionary
  - hip hop rhyming often defies mainstream pronunciations
  - want a language-independent model
- instead – identify rhyming lines using a **rhyme scheme detector** (Addanki & Wu, SLSP 2013)

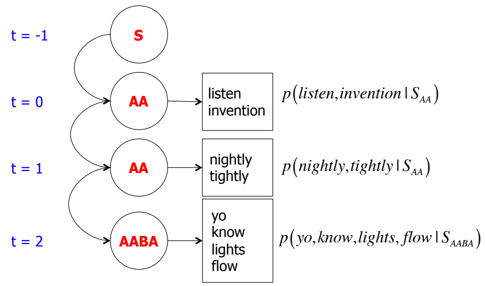
generative model for verses

- generated by a fully connected HMM

verse vs. stanza vs. line

line
All right stop collaborate and listen
Ice is back with my brand new invention
stanza AA
Something grabs a hold of me tightly
Flow like a harpoon daily and nightly
Will it ever stop yo, I don't know
Turn off the lights, and I'll glow
stanza AABA
To the extreme I rock a mic like a vandal
Light up a stage and wax a chump like a candle
verse
Dance go rush to the speaker that booms
I'm killing your brain like a poisonous mushroom
Deadly when I play a dope melody
Anything less than the best is a felony
Love it or leave it you better gain weight
You better hit bull's eye the kid don't play
If there was a problem, yo I'll solve it
Check out the hook while my DJ revolves it

rhyme scheme detection
(Addanki & Wu, SLSP 2013)



generative model for verses

- generated by a fully connected HMM
- each state S_r is a stanza with a particular rhyme scheme r
- emissions are a sequence of final tokens $x_{1\ldots n}$ in each line of the stanza
- a single textual line of lyrics might contain two lyrical lines separated by a comma

rhyme scheme detection
(Addanki & Wu, SLSP 2013)

- uses an HMM based generative model for verses
- no prior linguistic or phonetic information
- partitions each verse into stanzas with different rhyme schemes
- f-score – 44.06% as evaluated by humans

result — data selection
disfluency correction + rhyme scheme detection

model+disfluency strat.	fluency (good)	fluency (≥acceptable)	rhyming (good)	rhyming (≥acceptable)
PBSMT+filtering	**4.3%**	**13.72%**	3.53%	7.06%
PBSMT+correction	3.14%	4.70%	1.57%	4.31%
PBSMT+RS+filtering	**31.76%**	**43.91%**	12.15%	21.17%
PBSMT+RS+correction	30.59%	43.53%	1.96%	9.02%
TG+filtering	17.25%	46.27%	18.04%	33.33%
TG+correction	**21.18%**	**54.51%**	23.53%	39.21%
TG+RS+filtering	28.63%	56.86%	14.90%	34.51%
TG+RS+correction	**34.12%**	**60.39%**	20.00%	42.74%

- disfluency correction is better than disfluency filtering
  - improves both fluency and (surprisingly?) rhyming
- rhyme scheme detection
  - improves fluency for both TG and PBSMT models
  - improves the fraction of sentences with ≥ acceptable rhyming
  - similar results can be observed for ISTG models also (coming up)

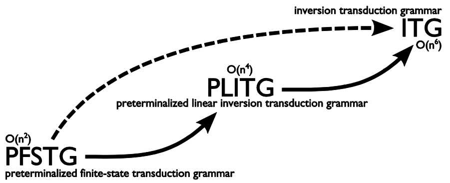
chunking vs segmentation?

transduction grammar induction

transduction grammar induction

- bracketing inversion transduction grammars (BITG) (Wu 1997)
  - empirically high coverage, accuracy across various NLP tasks
  - sufficient expressiveness to handle word associations between lines
  - efficient induction and decoding algorithms
- compare two approaches, trained on same amount of data
  - **TG** token-based BITG
  - **ISTG** interpolated segmental BITG
- token-based BITG captures word associations better
  - efficient induction algorithm using
    - EM
    - beam pruning
    - bootstrapped induction (Saers, Addanki & Wu, COLING 2012)
- segmental BITG captures phrasal associations better
  - efficient induction algorithm using
    - greedy iterative segmentation of transduction rules
    - MDL driven induction (Saers, Addanki & Wu, SSST 2013)

bottom-up transduction rule chunking
induction by bootstrapping (Saers, Addanki & Wu, COLING 2012)



token-based vs. segmental grammars

- transduction grammar rules can contain
  - **rules** $A \rightarrow [B\ A\ \text{“long”/“wrong”}]$
    $A \rightarrow \langle A\ \text{“felt bad”/“what I really had”}\ B \rangle$
  - **lexical rules** $A \rightarrow \text{“long”/“wrong”}$
    $A \rightarrow \text{“felt bad”/“what I really had”}$
  - **structural rules** $A \rightarrow [A\ A]$
    $A \rightarrow \langle A\ B \rangle$

token-based vs. segmental grammars

- transduction grammar rules can contain
  - **rules** $A \rightarrow [B\ A\ \text{“long”/“wrong”}]$
    $A \rightarrow \langle A\ \text{“felt bad”/“what I really had”}\ B \rangle$
  - **lexical rules** $A \rightarrow \text{“long”/“wrong”}$
    $A \rightarrow \text{“felt bad”/“what I really had”}$
  - **structural rules** $A \rightarrow [A\ A]$
    $A \rightarrow \langle A\ B \rangle$
- **token-based** transduction grammars
  - biterminals contain at most one token in each language
    $A \rightarrow e/f \ |\ e/\varepsilon \ |\ \varepsilon/f$
  - simple and efficient learning algorithms
  - suffer from a lack of fluency in the output

token-based vs. segmental grammars

- transduction grammar rules can contain
  - **rules** $A \rightarrow [B\ A\ \text{“long”/“wrong”}]$
    $\cancel{A \rightarrow \langle A\ \text{“felt bad”/“what I really had”}\ B \rangle}$
  - **lexical rules** $A \rightarrow \text{“long”/“wrong”}$
    $\cancel{A \rightarrow \text{“felt bad”/“what I really had”}}$
  - **structural rules** $A \rightarrow [A\ A]$
    $A \rightarrow \langle A\ B \rangle$
- **token-based** transduction grammars
  - biterminals contain at most one token in each language
    $A \rightarrow e/f \ |\ e/\varepsilon \ |\ \varepsilon/f$
  - simple and efficient learning algorithms
  - suffer from a lack of fluency in the output

token-based vs. segmental grammars

- transduction grammar rules can contain
  - **rules** $A \rightarrow [B\ A\ \text{“long”/“wrong”}]$
    $A \rightarrow \langle A\ \text{“felt bad”/“what I really had”}\ B \rangle$
  - **lexical rules** $A \rightarrow \text{“long”/“wrong”}$
    $A \rightarrow \text{“felt bad”/“what I really had”}$
  - **structural rules** $A \rightarrow [A\ A]$
    $A \rightarrow \langle A\ B \rangle$
- **segmental** transduction grammars
  - biterminals can contain <u>multiple</u> tokens in each language
    $A \rightarrow e_{0..1}/f_{0..2} \ |\ e_{0..1}/\varepsilon \ |\ \varepsilon/f_{0..2}$ (Wu 1997)
  - models phrases for more fluent output
  - but unsupervised learning algorithms not as well developed…

learning segmental bigrammars

- unfortunately there still aren’t many algorithms for segmental transduction grammar induction that
  - are purely unsupervised
  - assume the same matched training and testing models
- most SMT approaches instead
  - use heuristics to extract segments from token alignments
  - typ. justified by BLEU motivations not applicable to our task

segmental model

- segmental ITG learned using a top-down rule segmenting approach driven by a MAP/minimum description length objective (Saers, Addanki & Wu, IJCNLP 2013)

interpolated segmental model

- segmental ITG learned using a top-down rule segmenting approach driven by a MAP/minimum description length objective (Saers, Addanki & Wu, IJCNLP 2013)
- aim: balance fluency and expressivity by interpolating the token-based ITG G_a with a segmental ITG G_b
  - interpolation parameter α controls mixture
    $p_{a+b}(r) = \alpha p_a(r) + (1-\alpha) p_b(r)$

result — English hip hop
challenge-response test examples

challenge	man i **die** to **see** em all thun i just don’t **care**
FREESTYLE	in the **sky** and **me** the in pollan and the you **there**
PBSMT	man i live to see em all i just don’t care
challenge	did a twelve year **bid** in the streets and held it **down**
FREESTYLE	to the girls here **kid** the and to the thought the **now**
PBSMT	did a year in the streets and it down
challenge	oh i believe in **yesterday**
FREESTYLE	can you see the **day**
PBSMT	oh i believe in tomorrow
challenge	what would i **do**
FREESTYLE	just me and **you**
PBSMT	what would you do
challenge	cause you ain’t going **home** till the early **morn**
FREESTYLE	and the you this **alone** i i gotta **on**
PBSMT	cause you and your friends aint nothing but

result
learned transduction grammar rule examples

transduction grammar rule	*log prob.*
$A \rightarrow$ long/wrong	-11.6747
$A \rightarrow$ rhyme/time	-11.6604
$A \rightarrow$ felt bad/couldn't see what i really had	-11.3196
$A \rightarrow$ matter what you say/leaving anyway	-11.8792
$A \rightarrow$ arhythamatic/this rhythm is sick	-12.3492

result — token-based vs segmental
segmental TG improves fluency and rhyming

model	*fluency (≥good)*	*fluency (≥acceptable)*	*rhyming (≥good)*	*rhyming (≥acceptable)*
PBSMT	3.14%	4.70%	1.57%	4.31%
TG	21.18%	54.51%	23.53%	39.21%
ISTG	**26.27%**	**57.64%**	**27.45%**	**48.23%**
PBSMT+RS	30.59%	43.53%	1.96%	9.02%
TG+RS	**34.12%**	60.39%	20.00%	42.74%
ISTG+RS	30.98%	**61.18%**	**30.98%**	**53.72%**

- interpolated segmenting TG (ISTG) produces more fluent responses than token-based TG on both data sets (more later)
- ISTG also produces better rhyming responses (surprising?)
- results also demonstrate that off-the-shelf phrase-based SMT systems (PBSMT) cannot be directly adopted for this task

Maghrebi French hip hop

learning to freestyle in **Maghrebi French**

- advantage of our linguistics-lite model is that it should work independent of language
- test if it can learn to generate response lyrics in languages other than English
- initial experiments on Maghrebi French hip hop lyrics
- our model performs surprisingly well despite
  - no special adaptation
  - much smaller training data size
  - diverse and noisy training data

Maghrebi French hip hop corpus

- about 1300 hip hop song lyrics
- majority in **Maghrebi French**: French interspersed with romanized Arabic, Berber and English phrases
  - De la traversée du désert au bon couscous de **Yéma** (*Yéma = My mother)*
  - **a yemmi ino** = *my son* / **a thizizwith** = *a bee*
  - T'es **game over, game over** / Le son de **Chicken wings**
- linguistically complex – language dependent models will be hard to adapt
- 47,000 sentence pairs selected using rhyme scheme detection

result — Magrebi French hip hop
encouraging results despite sparse data

- FREESTYLE produces responses rated
  - *good* – 9.2% (fluency), 14.5% (rhyming)
  - *≥acceptable* – 30.2% (fluency), 38% (rhyming)
- encouraging results despite small and diverse training corpus causing sparse data issues
- error analysis on responses indicates
  - realization of less common rhyme schemes like **ABAB**
  - responses with semantically related terms

result — Magrebi French hip hop
learned transduction grammar rule examples

transduction grammar rule	*log prob.*
$A \rightarrow$ terre/la guerre	-9.4837
$A \rightarrow$ haine/peine	-9.77056
$A \rightarrow$ mal/pays natal	-10.6877
$A \rightarrow$ je frissonne/mi corazon	-11.0931
$A \rightarrow$ gratteurs/rappeurs	-11.7306

result — Magrebi French hip hop
challenge-response test examples

challenge	Si je me **trompe**
response	faut que je **raconte**
challenge	Un jour je suis un **livre**
response	et ce que je de **vivre**
challenge	Pacha mama ils ne voient pas ta **souffrance**
response	Combat ni leur de voulait de la **décadence**
challenge	le palestine n'etait pas une **terre** sans peuple
response	le **darfour d'autre** de la **guerre** on est
challenge	Une **banlieue** qui **meut**
response	les **yeux** et

conclusions

- FREESTYLE: first known model for learning how to rap battle
  - transduces challenge lyrics to improvised responses
  - transduction grammar induction is fully unsupervised
  - learns fluent, rhyming responses absent linguistic knowledge
- new MDL driven learning of hip hop ITG rules by segmentation
  - segmental grammars improve on token-based grammars
  - both outperform off-the-shelf PBSMT contrastive baseline
- hip hop domain specific models against very noisy training data yield more fluent and better rhyming responses
  - data selection via unsupervised rhyme scheme detection model
  - disfluency correction
- completely unsupervised generation of hip hop challenge responses without linguistic knowledge despite the noisy domain
- generalizes to non-English hip hop: encouraging results on Maghrebi French validate language independence assumptions

maximum *a posteriori* objective

$$\underset{\Phi}{\operatorname{argmax}} P(\Phi)\, P(D \mid \Phi)$$

prior

bilingual corpus likelihood

$$P(\Phi) = P(\Phi_G)\, P(\Phi_S \mid \Phi_G) P(\theta_\Phi \mid \Phi_S, \Phi_G)$$

MTG

DL prior*

Dirichlet prior

$$P(\theta_\Phi \mid \Phi_S, \Phi_G) = \prod_{i=0}^{N-1} \frac{1}{B(\alpha_0, \alpha_1, \ldots, \alpha_{R_{n_i}-1})} \prod_{j=0}^{R_{n_i}-1} \theta_\Phi^{n_i}(j)$$

$$-\log_2(P(\Phi_S \mid \Phi_G)) \propto \mathrm{DL}(\Phi_S)$$

putting everything together, we want

$$\underset{\Phi_G, \Phi_S, \theta_\Phi}{\operatorname{argmax}} P(\Phi_G)\, P(\Phi_S \mid \Phi_G) P(\theta_\Phi \mid \Phi_S, \Phi_G) P(D \mid \Phi_G, \Phi_S, \theta_\Phi)$$

*prior constructed in terms of the grammar's description length

challenges with symbolic ITGs

- unsupervised ITG induction remains hard
  - extremely large model space!
- nonterminals don't capture context efficiently
  - intractable – each context has explicit nonterminal
  - solution: replace nonterminals with feature vectors
- new idea: apply **TRAAM**, a distributed representation for ITGs we began developing for statistical MT (Addanki & Wu 2014)
  - *bilingual* recursive neural network model
  - uses *both* input & output language contexts

modeling recursive structures

- TRAAM goes beyond neural network approaches that model **monolingual recursive structures**
- neural language models and SRNs (Bengio *et al.* 2003)
  - contextual history modeled by a RNN
- convolutional networks (Collobert & Weston 2008)
  - learn vector representations of words
  - used in NLP tasks such as POS tagging, chunking and SRL
- RAAM and recursive autoencoders (RAE) (Pollack 1990, Socher *et al.* 2011)
  - can be more flexible than convolutional networks
  - RAEs have been successfully applied to sentiment prediction

bilingual vector space models

- dearth of vector space models for **compositional** learning of bilingual relations
- predominantly augment "shake-n-bake" SMT modeling assumptions using feature vectors
- n-gram translation models (Son et al., 2012)
  - bilingual generalization of class based n-grams using distributed representations
  - fails to model compositionality and cross-lingual reordering
- bilingual word embeddings (Zou et al., 2013)
  - recurrent NNLM model with SMT word alignments
  - only learns non-compositional features

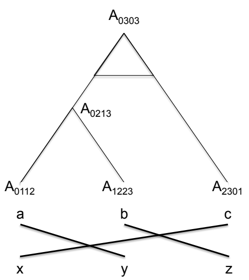
bilingual vector space models

- NNLMs + input language context (*Devlin et al.* 2014)
  - does not model input and output language features simultaneously
- recurrent probabilistic models (Kalchbrenner & Blunsom 2013)
  - generates an input sentence representation that generates an output sentence
  - lacks structural constraints and relies on a LM to reorder output
- reordering prediction using RAEs (Li *et al.* 2013)
  - **monolingual** RAEs to predict reordering in a maxent ITG model
  - uses only input language context

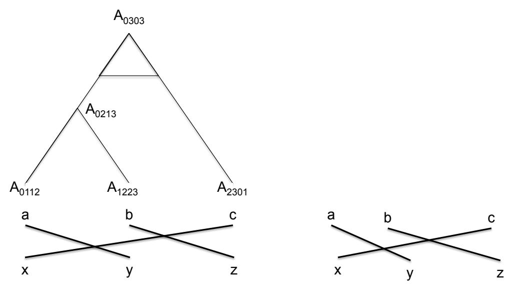
TRAAM
model definition

- bilingual recursive neural network
  - fully bilingual generalization of monolingual RAAM model
  - vectors represent **bilingual** constituents
- uniform feature vector dimension
  - task-dependent representation learning
  - similar biconstituents have similar vectors
  - feature vector clusters represent **soft bilingual categories**
- generates feature vectors recursively from smaller biconstituents
  - language bias via dimensionality reduction

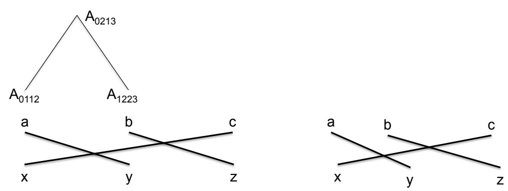
BITG → TRAAM



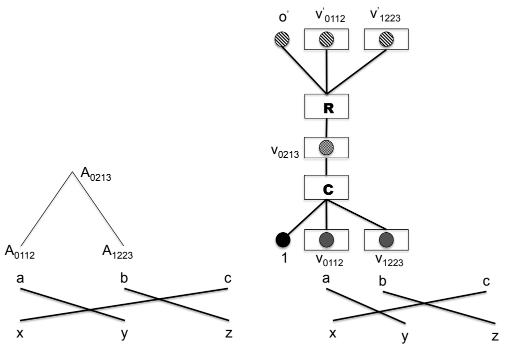
BITG → TRAAM



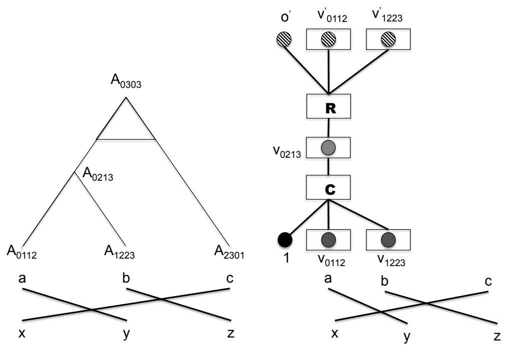
BITG → TRAAM



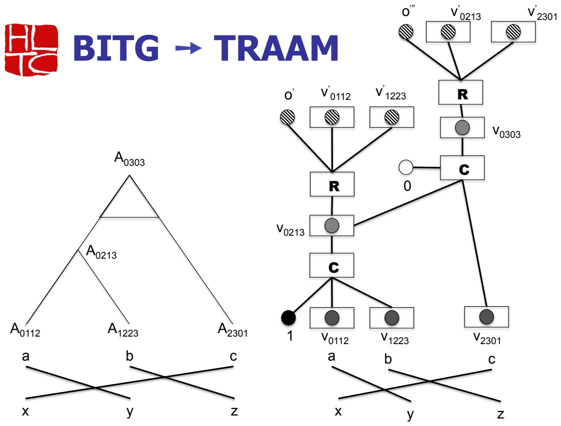
BITG → TRAAM



BITG → TRAAM



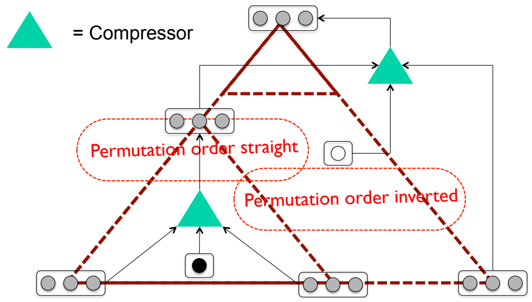
BITG → TRAAM



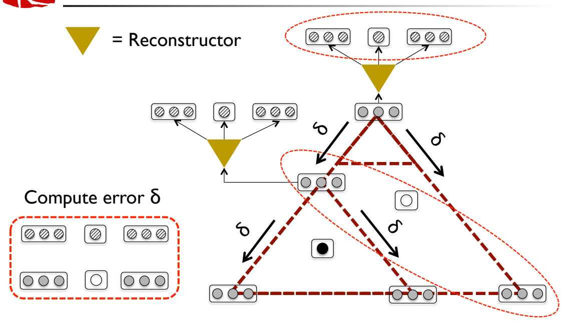
TRAAM
model training

- TRAAM network contains a compressor and a reconstructor network
  - generalizes RAAM to represent **bilingual** sequences
  - for **bracketing** ITGs, reordering can be represented by a single bit (straight vs. inverted)
- biparses from a BITG are used to learn the network weights
  - compressor network generates feature vectors recursively
  - objective – to ensure the context of children is captured efficiently
  - reconstructor network provides the loss function
- backpropagation with structure is used to compute gradients
  - L-BFGS can be used to optimize network weights (Goller & Kulcher 1996)

TRAAM forward propagation



TRAAM backpropagation



TRAAM
improvisation using transduction engine

- reconstruction error used as a feature for transduction (decoding)
  - provides the context that symbolic ITGs lack
  - used in a log-linear combination with grammar and LM score
- transduction heuristics similar to Wu *et al.* (2013) also applied
  - rules mapping a surface form to itself were penalized
  - singleton rules were penalized
- our ITG transducer (decoder) is capable of handling features
  - cube pruning (Chiang 2008) used to generate k-best hypotheses
  - feature weights were determined via manual inspection on a small development set

challenges in flamenco

- no clear boundary between music and dance
- "constrained improvisation"
- regular and irregular hypermetrical structures
- rapid switching between 3/4 and 6/8 meters
- heavy syncopation
- sudden, misleading off-beat accents and patterns
- frequent eliding of downbeat accents (which humans and automati meter-finding algorithms typically rely on)
- **expert musicians rely on**
  - complex hypermetrical knowledge
  - syncopated meter patterns
  - irregular real

to dynamically recognize when to switch meters/patterns

learning flamenco hypermetrical structure

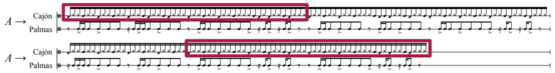
- simultaneous learning of
  - metrical structure
  - hypermetrical structure
  - multipart structural relations
- learn the relationship between parallel frames of reference

(Wu, ISMIR 2013, SMPC 2015)

learning flamenco hypermetrical structure

initial transduction grammar is just the parallel training corpus

$S \rightarrow A$
$A \rightarrow A\ A$



learning flamenco hypermetrical structure

initial transduction grammar is just the parallel training corpus

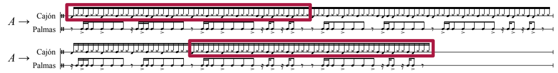
$S \rightarrow A$
$A \rightarrow A\ A$

1

learning flamenco hypermetrical structure

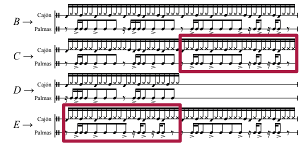
initial transduction grammar is just the parallel training corpus

$S \rightarrow A$
$A \rightarrow A\ A$



1

$S \rightarrow A$
$A \rightarrow A\ A$
$A \rightarrow B\ C$
$A \rightarrow D\ E$



learning flamenco hypermetrical structure

$S \rightarrow A$
$A \rightarrow A\ A$
$A \rightarrow B\ C$
$A \rightarrow D\ E$

learning flamenco hypermetrical structure

2

$S \rightarrow A$
$A \rightarrow A\ A$
$A \rightarrow B\ C$
$A \rightarrow D\ E$

learning flamenco hypermetrical structure

$S \rightarrow A$
$A \rightarrow A\ A$
$A \rightarrow B\ C$
$A \rightarrow D\ E$
$C \rightarrow F\ G$
$E \rightarrow G\ H$

2

$S \rightarrow A$
$A \rightarrow A\ A$
$A \rightarrow B\ C$
$A \rightarrow D\ E$

learning flamenco hypermetrical structure

$S \rightarrow A$
$A \rightarrow A\ A$
$A \rightarrow B\ C$
$A \rightarrow D\ E$
$C \rightarrow F\ G$
$E \rightarrow G\ H$

learning flamenco hypermetrical structure

$S \rightarrow A$
$A \rightarrow A\ A$
$A \rightarrow B\ C$
$A \rightarrow D\ E$
$C \rightarrow F\ G$
$E \rightarrow G\ H$

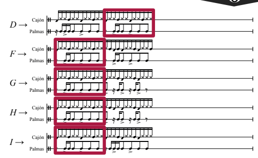
3

learning flamenco hypermetrical structure

$S \rightarrow A$
$A \rightarrow A\ A$
$A \rightarrow B\ C$
$A \rightarrow D\ E$
$C \rightarrow F\ G$
$E \rightarrow G\ H$

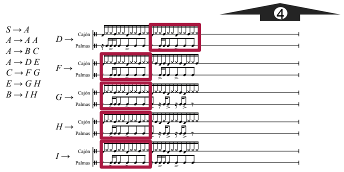
3

$S \rightarrow A$
$A \rightarrow A\ A$
$A \rightarrow B\ C$
$A \rightarrow D\ E$
$C \rightarrow F\ G$
$E \rightarrow G\ H$
$B \rightarrow I\ H$

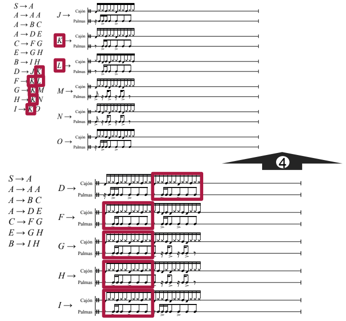


learning flamenco hypermetrical structure

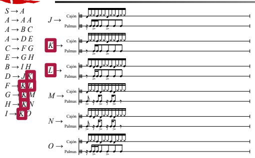
learning flamenco hypermetrical structure



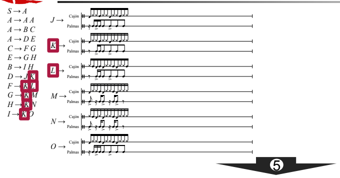
learning flamenco hypermetrical structure



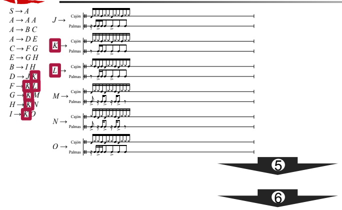
learning flamenco hypermetrical structure



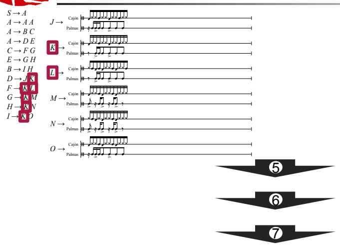
learning flamenco hypermetrical structure



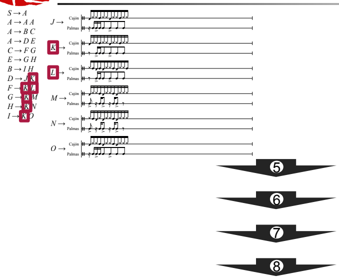
learning flamenco hypermetrical structure



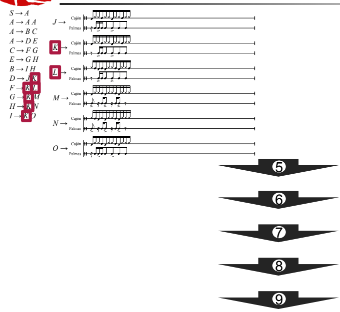
learning flamenco hypermetrical structure



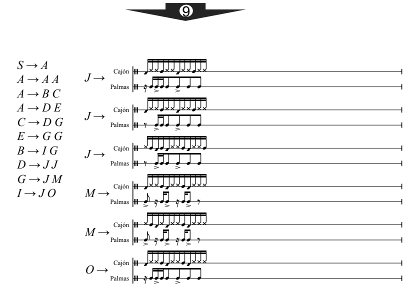
learning flamenco hypermetrical structure



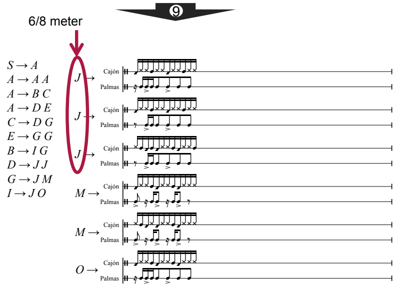
learning flamenco hypermetrical structure



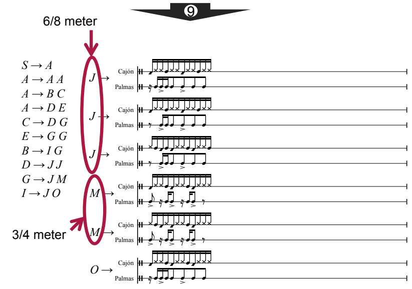
learning flamenco hypermetrical structure



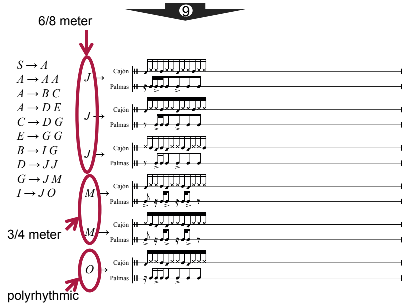
learning flamenco hypermetrical structure



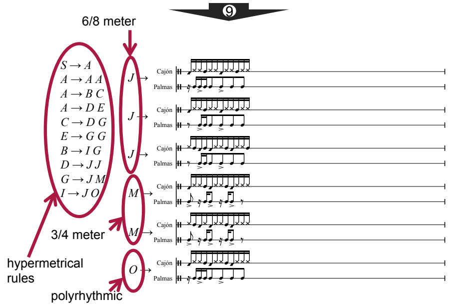
learning flamenco hypermetrical structure



learning flamenco hypermetrical structure



learning flamenco hypermetrical structure



escaping from blocks world

The same exact core capability of transduction grammar induction and application appears effective not only for conventional AI tasks, but also for all sorts of creative tasks.

ESSCaSS'15 day 2 Nelijärve, Estonia 2015.08.19

AI = Learning to Translate
Language, Music, and Creativity

Dekai Wu
dekai@cs.ust.hk http://www.cs.ust.hk/~dekai

HKUST
Human Language Technology Center
Department of Computer Science and Engineering
University of Science and Technology, Hong Kong