

slides

Corpus-Based Lexical Choice in Natural Language Generation

Srinivas Bangalore and Owen Rambow

{srimi,rambow}@research.att.com

AT&T Labs – Research

Overview

- The FERGUS Natural Language Generation System
- Representing Meaning
- Architecture: Lexical and Syntactic Choice
- Choosing Lexemes in a Tree
- Conclusion

Stochastic Models in Realization

- FERGUS performs (some) lexical choice, (some) syntactic choice, morphology, linearization
- Useful if:
 - Need to generate wide variety of texts (MT)
 - Need to quickly customize generator to new domains or genres

Stochastic Models in Generation: Recent Approaches

- Langkilde and Knight (INLG 1998, NAACL 2000)

Features:

- Generation from “semantic” interlingua for MT
- Hand-coded rules overgenerate possible realizations for interlingua elements (including multiple target language lexemes)
- Rule output composed into lattice
- Linear Language Model (bigram) chooses best path through lattice

Problem:

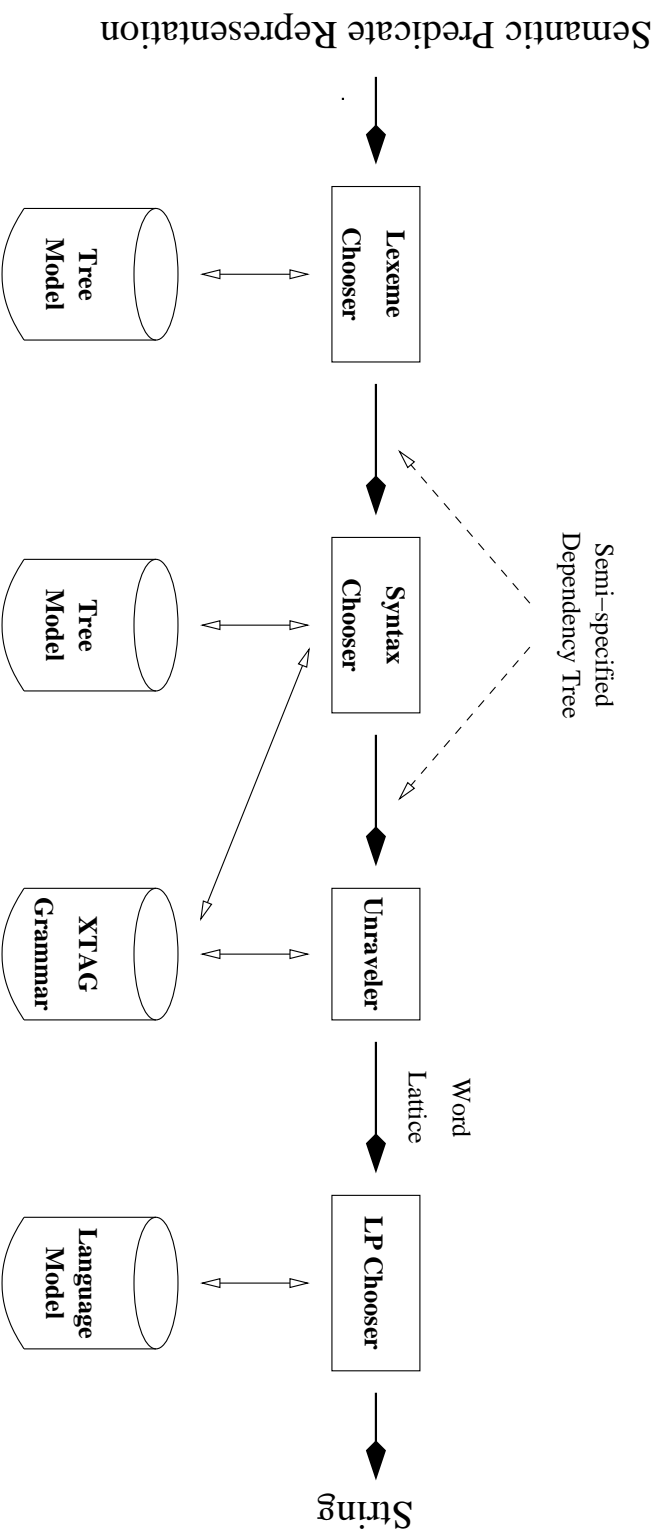
- Syntactic locality does not translate into string-adjacency

Stochastic Models in Generation: Our Approach

- Corpus-based learning to refine syntactic and lexical choice; generate surface string
- Use of stochastic tree model *and* linear language model
- Use of pre-existing hand crafted grammar: XTAG

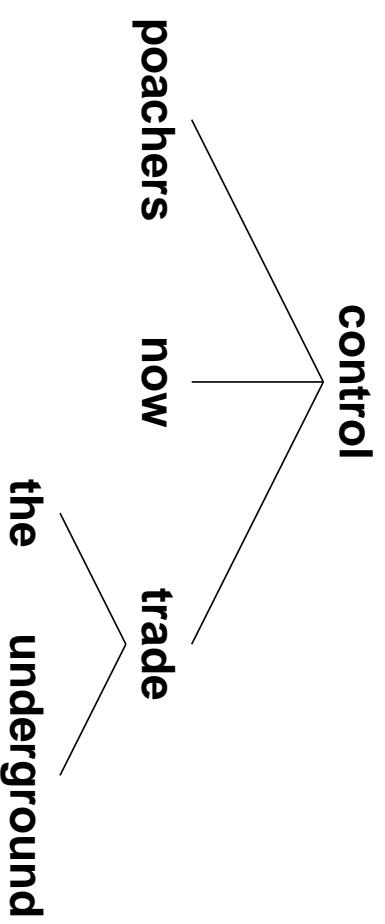
FERGUS System Architecture

- FERGUS: Flexible Empirical/Rationalist Generation Using Syntax



Input Representation

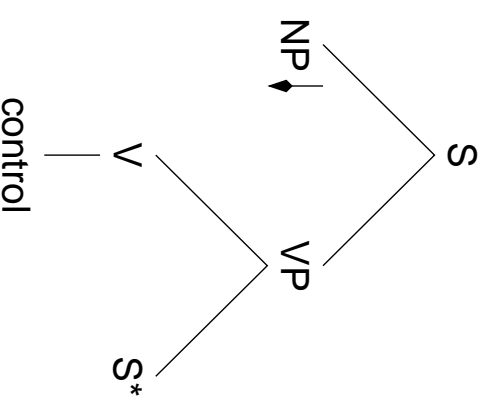
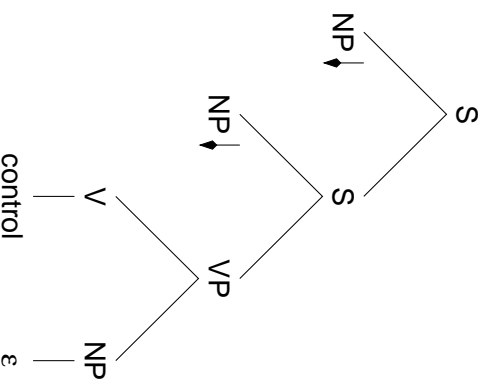
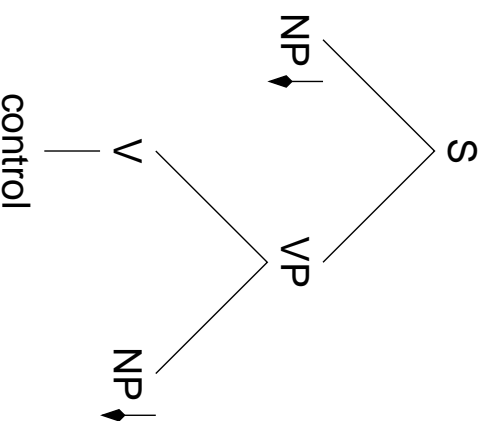
- Dependency tree
 - any feature can be specified



Syntax Chooser

Representin Syntax

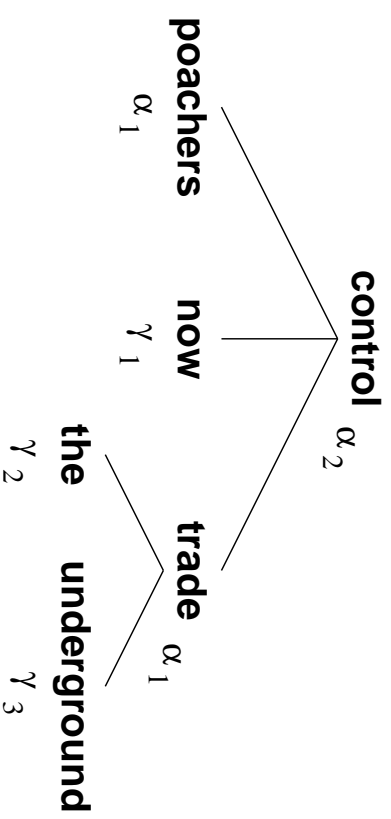
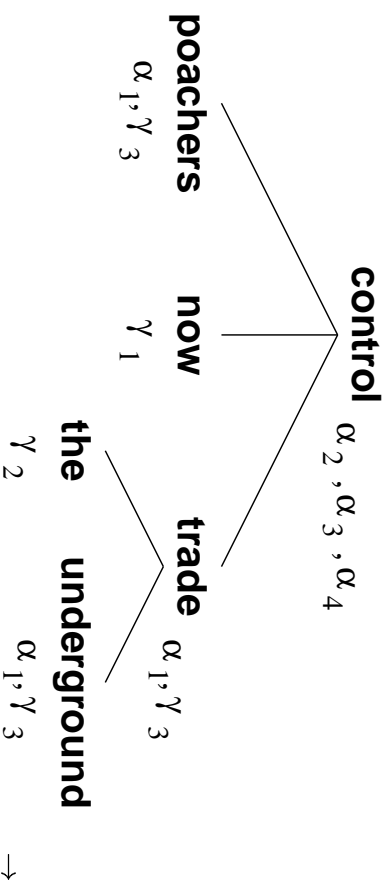
- Supertag: Elementary trees of Lexicalized Tree-Adjoining Grammar
 - grammatical function
 - subcategorization frame (active valency)
 - passive valency with direction of modification
 - realizations of arguments (passive, wh-movement, relative clause)



Syntax Chooser

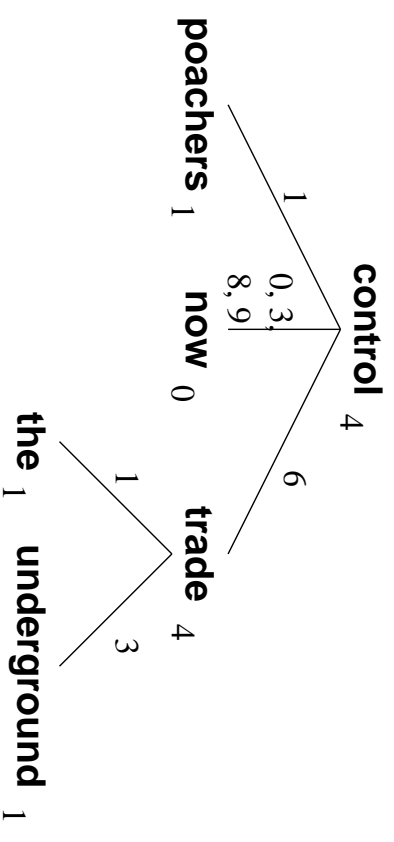
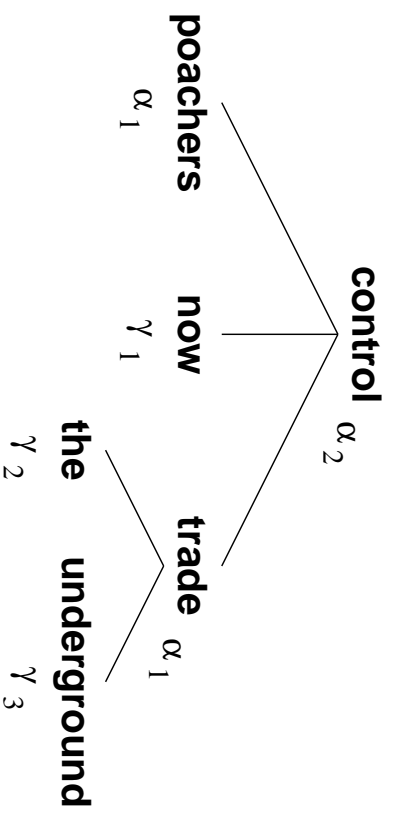
Choosing Syntax

- Given a dependency tree, assign the appropriate supertag to each node



Unraveler

- Dependency tree + supertags = semi-specified derivation tree
 - Argument positions need not be fixed in input
 - Adjunction sites of γ -trees not specified
- Given a semi-specified derivation tree, produce a word lattice
- XTAG grammar specifies possible positions of daughters in mother supertag.



LP Chooser

- Given a word lattice produce the most likely string
- A stochastic n -gram model is
 - used to rank the likelihood of the strings in the lattice
 - represented as a weighted finite-state automaton
 - composed with the word lattice
- Retrieves the best path through the composed lattice

BestPath (Lattice o LangModel)

Representing Meaning

- Generation : transforming concepts to language
- Stochastic generation : learn mapping (concepts → language) from corpus
- Problem: how to represent concepts?
- Idea: use WordNet synonym sets to represent lexical meaning
- Example: synsets for verb *fear*

Sense	Synset
1	fear, dread
2	reverence, fear, revere, venerate
3	fear
...	...

Representing Meaning (2)

- Idea: use WordNet synonym sets to represent lexical meaning
- Problem: currently no corpus of lexemes annotated with synsets
- Solution: use corpus annotated with *meaning potential* of lexeme = set of all possible synonyms of lexeme
- Example: Supersynset for verb *fear*

fear, dread, reverence, revere, venerate
--
- Advantage: do not need annotated corpus
- Note: can also use any other set as “meaning potential”

Representing Meaning

- Original task: choice of lexeme to convey lexical meaning
- New task: choice of synonymous lexeme for lexeme (or choice of lexeme from given set)
- Useful: when lexical variation is desired but not understood
the themes of liberty and freedom of expression
- Useful: in machine translation

English: wear a hat/shoes

UI dài màozi 'wear hat'

ci chuān xiézi 'wear shoes'

Experimental Setup

- Lexical choice task: take dependency tree labeled with lexemes, return tree labeled with “better” lexemes (from supersynsets)
- Need to empirically investigate: architecture of generator; factors in lexical choice
- Training corpus: One million words of WSJ corpus
- Test corpus:
 - 100 randomly chosen sentences
 - average sentence length 16.7 words
 - average ambiguity: 4.8 synonyms per lexeme
- Use random choice as baseline

Experimental Setup (2)

- Lexical choice task: take dependency tree labeled with lexemes, return tree labeled with “better” lexemes
- Issue: since corpus is labeled with lexemes (not meanings), input tree already contains optimal lexemes!
- Solution: use random choice as final backoff (e.g., for completely unseen words)
- Use completely random choice as baseline

Evaluation Metric

- Complex issue
- Metric
 - objective and automatic
 - without human intervention
 - quick turnaround
- These metrics not designed to compare realizers (but . . .)
- See paper at INLG 2000 conference (Bangalore & al 2000)

Two Evaluation Metrics

- **Bag Accuracy** = $\frac{\text{correct lexemes}}{\text{all lexemes}}$
- **String Accuracy** = $(1 - \frac{M+I'+D'+S}{R})$
 - String edit distance between reference string and result string (length in words: R)
 - * Substitutions (S)
 - * Insertions (I)
 - * Deletions (D)
 - * Moves = pairs of Deletions and Insertions (M)
 - * Remaining Insertions (I') and Deletions (D')

– Example:

	poachers	now	control	the	underground	trade	
trade	poachers			the	rebel		control
i	.	d	d	.	s	s	i

Architecture of Lexical Choice

Three Possible Architectures

- Lexical choice before syntactic choice (**L-S**)
- Lexical choice at same time as syntactic choice (**L+S**)
- Lexical choice after syntactic choice (**S-L**) picture?

Architecture of Generation

Results:

Model	String Accuracy	Bag Accuracy
random	0.34	0.69
L-S	0.62	0.87
L+S	0.61	0.87
S-L	0.70	0.93

- Note: for **S-L**, lexical choice in linear language model slightly outperforms lexical choice in tree model

Architecture of Lexical Choice

Notes on **S-L** Architecture

- **S-L** different task from **L-S** and **L+S**:
 - Lexical choice is more constrained
 - Requires single lexeme in input, not set
 - In experiment, syntax chosen based on “best” lexeme

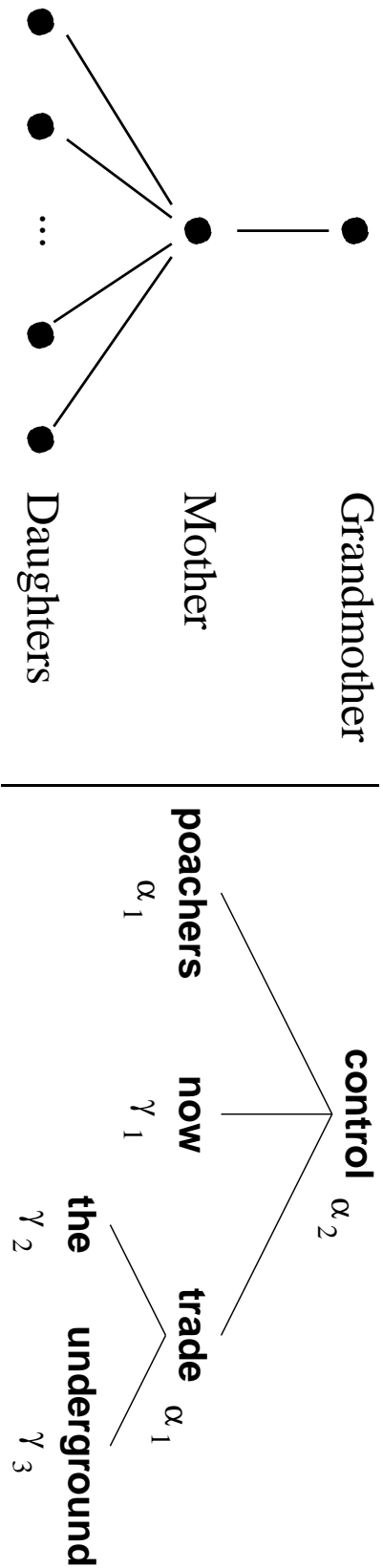
Thus, we expect better results for **S-L** than for **L-S** or **L+S**

- Do not choose **S-L** because it is unrealistic in general
- Note: If semantics determined syntax, then this option more appealing, but not so:
 - John gave/donated a car to the church
 - John gave/*donated the church a car

Models for Lexical Choice (L-S Task)

Models:

- Unigram: $argmax_{l \in L} p(l)$
- Mother-bigram: $argmax_{l \in L} p(l | l_m)$; backoff to unigram
- Daughters-bigram: $argmax_{l \in L} p(l | T_l) * p(T_l)$
Treelet (T_l) is modeled as a set of independent mother-daughter links.
- Mother-daughters-bigram: $argmax_{l \in L} p(l | T_l) * p(T_l) * p(l | l_m)$



Models for Lexical Choice (L-S Task)

Results:

Model	String Accuracy	Bag Accuracy
Random	0.34	0.69
Unigram	0.59	0.85
Mother-bigram	0.62	0.87
Daughters-bigram	0.63	0.88
Mother-daughters-bigram	0.67	0.90

- Note: Mother-bigram is previous **S-L** architecture
- Conclusion: increasing context in tree helps

Conclusion

- FERGUS uses stochastic models for lexical and syntactic choice
- Lexical choice before syntactic choice performs as well as combined lexical-syntactic choice
- Lexical choice after syntactic choice: different type of task
- Using more context (mother node, daughter nodes) improves lexical choice
- Useful for rapid development, for wide-coverage systems (MT)
- Need sense-tagged corpora!