Synergies in Language Acquisition

Mark Johnson

joint work with Katherine Demuth, Michael Frank,
Sharon Goldwater, Tom Griffiths and Bevan Jones

Macquarie University
Sydney, Australia

MACQUARIE UNIVERSITY

Outline

MACQUARIE UNIVERSITY

The Janus-faced character of modern computational linguistics

- *Engineering applications* (Natural Language Processing):
  - information extraction from large text collections
  - machine translation
  - speech recognition (?)
- *Scientific side* (Computational Linguistics):
  - computation is the *manipulation of meaning-bearing symbols* in ways that respect their meaning
  - studies language comprehension, production and *acquisition* as *computational processes*
- Spectacular (and lucrative) advances in NLP; can they help us *understand* language?
  - steam engines invented a century before thermodynamics and statistical mechanics

How can computational models help us understand language acquisition?

- Most computational linguistics research focuses on parsing or learning *algorithms*
- A *computational model* (Marr 1982) of acquisition specifies:
  - the input (information available to learner)
  - the output (generalisations learner can make)
  - a model that relates input to output
- This talk compares:
  - *staged learning*, which learns one kind of thing at a time, and
  - *joint learning*, which learns several kinds of things simultaneously,

  and demonstrates *synergies in acquisition* that only joint learners exploit
- We do this by *comparing models that differ solely in the kinds of generalisations they can form*

Bayesian learning as an "ideal observer" theory of learning

$$\underbrace{\mathrm{P}(\mathrm{Grammar} \mid \mathrm{Data})}_{\text{Posterior}} \propto \underbrace{\mathrm{P}(\mathrm{Data} \mid \mathrm{Grammar})}_{\text{Likelihood}} \underbrace{\mathrm{P}(\mathrm{Grammar})}_{\text{Prior}}$$

- Likelihood measures *how well grammar describes data*
- Prior expresses knowledge of grammar before data is seen
  - can be very specific (e.g., Universal Grammar)
  - can be very general (e.g., prefer shorter grammars)
- Prior can also express *markedness preferences* ("soft universals")
- Posterior is a *product* of both likelihood and prior
  - a grammar must do well on both to have high posterior probability
- Posterior is a *distribution* over grammars
  - captures *learner's uncertainty* about which grammar is correct

The acquisition of the lexicon as non-parametric inference

- What has to be learned in order to learn a word?
  - pronunciation (sequence of phonemes)
  - syntactic properties
  - semantic properties (what kinds of things it can refer to)

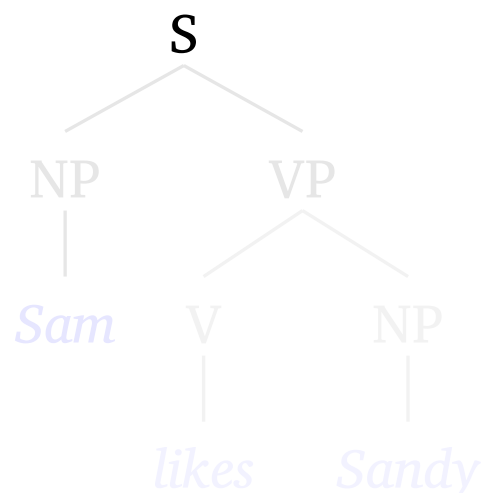
  There are *unboundedly many* different possible pronunciations (and possible meanings?)
- **Parametric inference:** learn values of a *finite number* of parameters
- **Non-parametric inference:**
  - possibly infinite number of parameters
  - learn which parameters are relevant as well as their values
- *Adaptor grammars* use a grammar to generate parameters for learning (e.g., possible lexical items)
  - builds on *non-parametric hierarchical Bayesian inference*

Outline

Probabilistic context-free grammars

- Probabilistic context-free grammars (PCFGs) define *probability distributions over trees*
- Each *nonterminal node* expands by
  - choosing a rule expanding that nonterminal, and
  - recursively expanding any nonterminal children it contains
- Probability of tree is *product of probabilities of rules* used to construct it

Probability θ_r	*Rule r*
1	S $\rightarrow$ NP VP
0.7	NP $\rightarrow$ *Sam*
0.3	NP $\rightarrow$ *Sandy*
1	VP $\rightarrow$ V NP
0.8	V $\rightarrow$ *likes*
0.2	V $\rightarrow$ *hates*

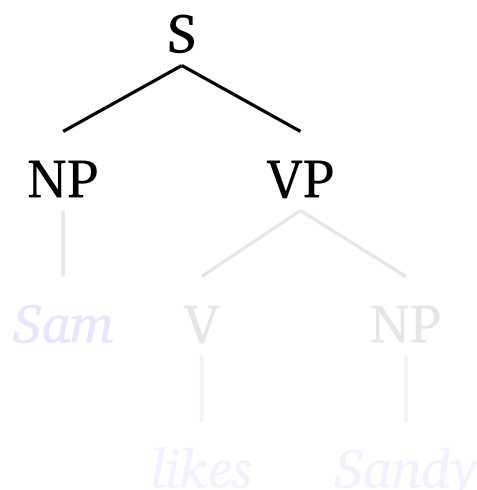


$$\mathrm{P}(\text{Tree}) = 1 \times 0.7 \times 1 \times 0.8 \times 0.3$$

Probabilistic context-free grammars

- Probabilistic context-free grammars (PCFGs) define *probability distributions over trees*
- Each *nonterminal node* expands by
  - choosing a rule expanding that nonterminal, and
  - recursively expanding any nonterminal children it contains
- Probability of tree is *product of probabilities of rules* used to construct it

Probability θ_r	*Rule* r
1	S $\rightarrow$ NP VP
0.7	NP $\rightarrow$ *Sam*
0.3	NP $\rightarrow$ *Sandy*
1	VP $\rightarrow$ V NP
0.8	V $\rightarrow$ *likes*
0.2	V $\rightarrow$ *hates*

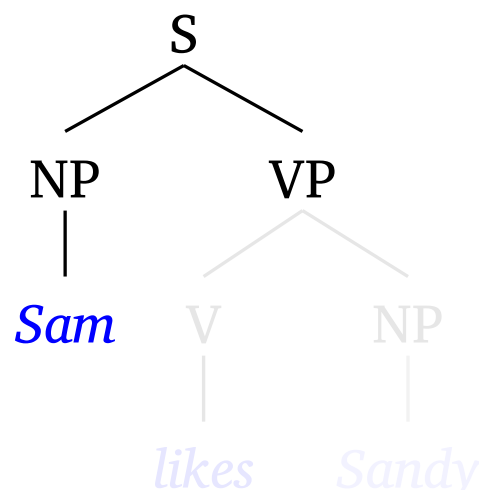


$$\mathrm{P}(\text{Tree}) = 1 \times 0.7 \times 1 \times 0.8 \times 0.3$$

Probabilistic context-free grammars

- Probabilistic context-free grammars (PCFGs) define *probability distributions over trees*
- Each *nonterminal node* expands by
  - choosing a rule expanding that nonterminal, and
  - recursively expanding any nonterminal children it contains
- Probability of tree is *product of probabilities of rules* used to construct it

Probability θ_r	*Rule* r
1	S $\rightarrow$ NP VP
0.7	NP $\rightarrow$ *Sam*
0.3	NP $\rightarrow$ *Sandy*
1	VP $\rightarrow$ V NP
0.8	V $\rightarrow$ *likes*
0.2	V $\rightarrow$ *hates*

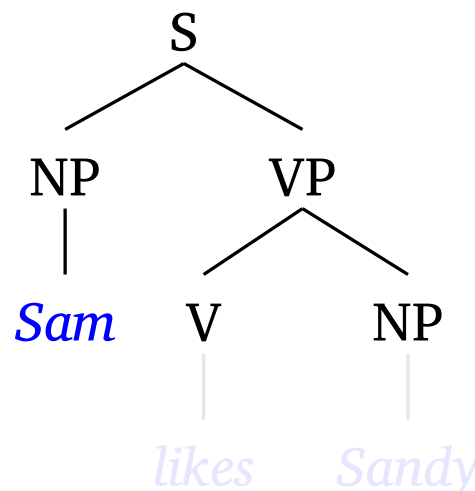


$$\mathrm{P}(\text{Tree}) = 1 \times 0.7 \times 1 \times 0.8 \times 0.3$$

Probabilistic context-free grammars

- Probabilistic context-free grammars (PCFGs) define *probability distributions over trees*
- Each *nonterminal node* expands by
  - choosing a rule expanding that nonterminal, and
  - recursively expanding any nonterminal children it contains
- Probability of tree is *product of probabilities of rules* used to construct it

Probability θ_r	*Rule* r
1	S $\rightarrow$ NP VP
0.7	NP $\rightarrow$ *Sam*
0.3	NP $\rightarrow$ *Sandy*
1	VP $\rightarrow$ V NP
0.8	V $\rightarrow$ *likes*
0.2	V $\rightarrow$ *hates*

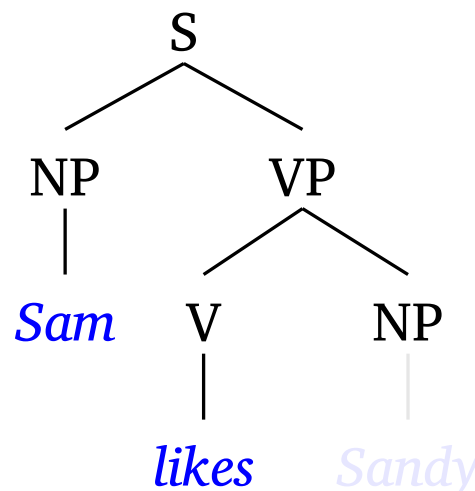


$$\mathrm{P}(\text{Tree}) = 1 \times 0.7 \times 1 \times 0.8 \times 0.3$$

Probabilistic context-free grammars

- Probabilistic context-free grammars (PCFGs) define *probability distributions over trees*
- Each *nonterminal node* expands by
  - choosing a rule expanding that nonterminal, and
  - recursively expanding any nonterminal children it contains
- Probability of tree is *product of probabilities of rules* used to construct it

Probability θ_r	*Rule r*
1	S $\rightarrow$ NP VP
0.7	NP $\rightarrow$ *Sam*
0.3	NP $\rightarrow$ *Sandy*
1	VP $\rightarrow$ V NP
0.8	V $\rightarrow$ *likes*
0.2	V $\rightarrow$ *hates*

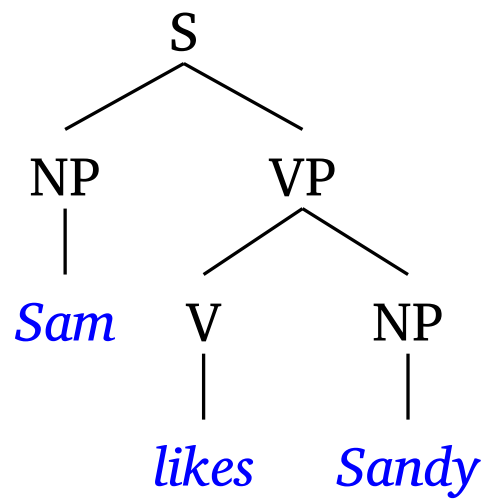


$$\mathrm{P}(\text{Tree}) = 1 \times 0.7 \times 1 \times 0.8 \times 0.3$$

Probabilistic context-free grammars

- Probabilistic context-free grammars (PCFGs) define *probability distributions over trees*
- Each *nonterminal node* expands by
  - choosing a rule expanding that nonterminal, and
  - recursively expanding any nonterminal children it contains
- Probability of tree is *product of probabilities of rules* used to construct it

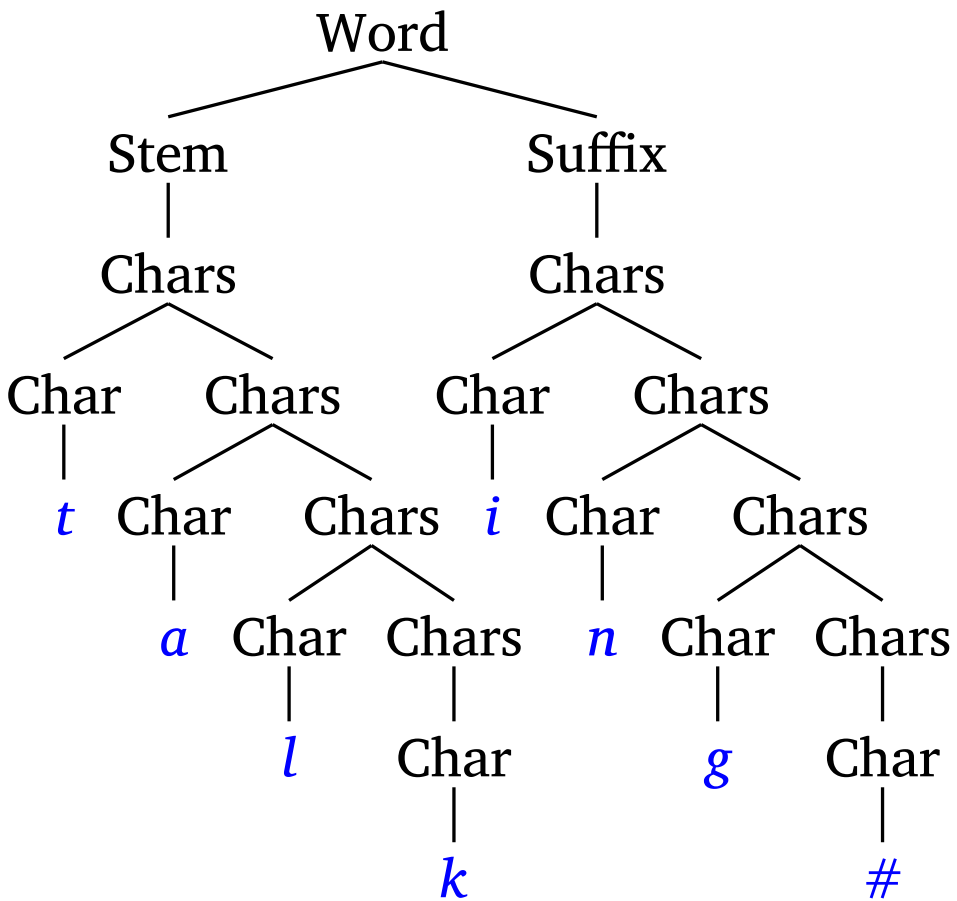
Probability θ_r	*Rule r*
1	S $\rightarrow$ NP VP
0.7	NP $\rightarrow$ *Sam*
0.3	NP $\rightarrow$ *Sandy*
1	VP $\rightarrow$ V NP
0.8	V $\rightarrow$ *likes*
0.2	V $\rightarrow$ *hates*



$$\mathrm{P}(\text{Tree}) = 1 \times 0.7 \times 1 \times 0.8 \times 0.3$$

A CFG for stem-suffix morphology

Word	→	Stem Suffix	Chars	→	Char
Stem	→	Chars	Chars	→	Char Chars
Suffix	→	Chars	Char	→	a | b | c | . . .



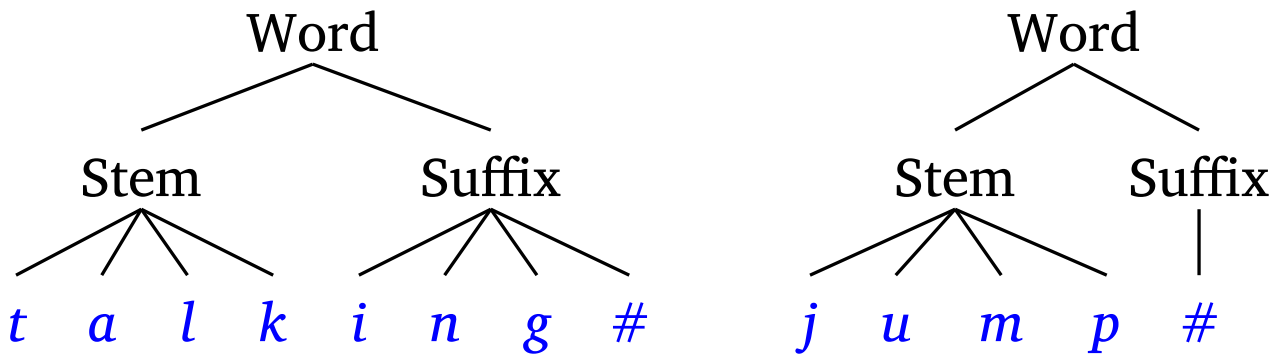
- Grammar's trees can represent any segmentation of words into stems and suffixes

⇒ Can *represent* true segmentation

- But grammar's *units of generalization (PCFG rules) are "too small"* to learn morphemes

A "CFG" with one rule per possible morpheme

$$\begin{aligned} \text{Word} &\rightarrow \text{Stem Suffix} \\ \text{Stem} &\rightarrow \textit{all possible stems} \\ \text{Suffix} &\rightarrow \textit{all possible suffixes} \end{aligned}$$



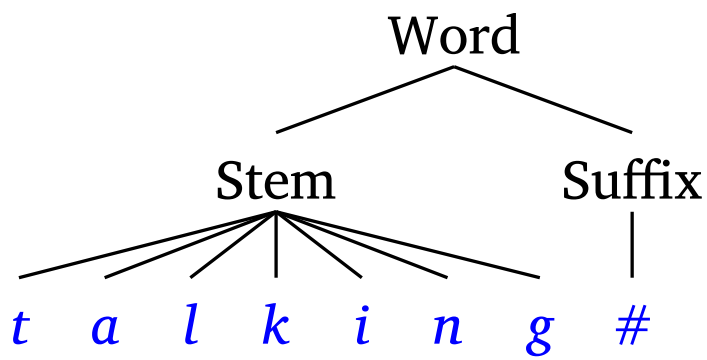
- A rule for each morpheme
  $\Rightarrow$ "PCFG" can represent probability of each morpheme
- *Unbounded number of possible rules, so this is not a PCFG*
  - not a practical problem, as only a finite set of rules could possibly be used in any particular data set

Maximum likelihood estimate for θ is trivial

- Maximum likelihood selects θ that minimizes KL-divergence between model and training data $\boldsymbol{W}$ distributions
- *Saturated model* in which each word is generated by its own rule replicates training data distribution $\boldsymbol{W}$ exactly

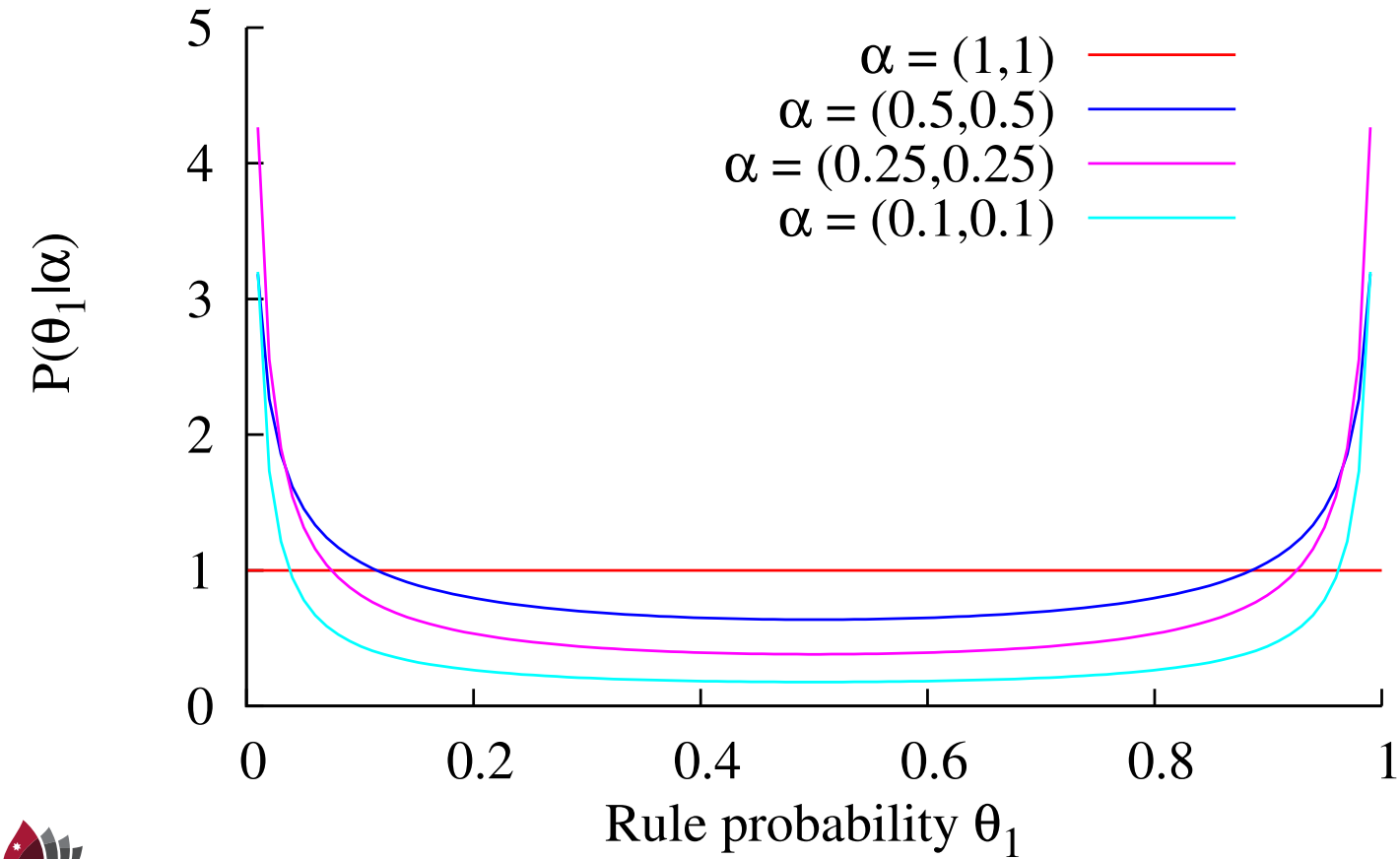
$\Rightarrow$ Saturated model is maximum likelihood estimate

- Maximum likelihood estimate does not find any suffixes



Forcing generalization via sparse priors

- Idea: use Bayesian prior that prefers fewer rules
- Set of rules is fixed in standard PCFG estimation, but can "turn rule off" by setting $\theta_{A \to \beta} \approx 0$
- Dirichlet prior with $\alpha_{A \to \beta} \approx 0$ prefers $\theta_{A \to \beta} \approx 0$



MACQUARIE UNIVERSITY

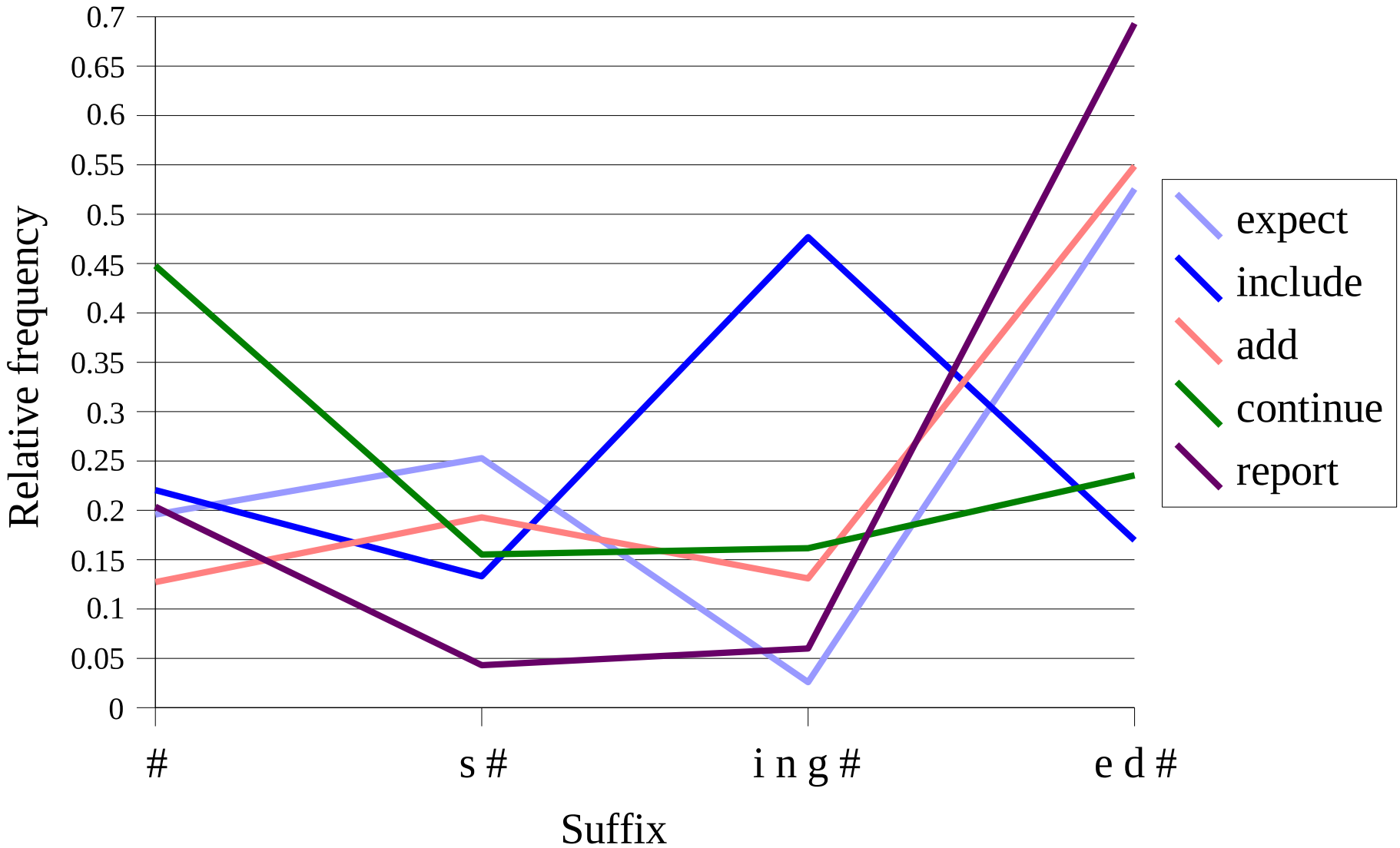
Morphological segmentation experiment

- Trained on orthographic verbs from U Penn. Wall Street Journal treebank
- Uniform Dirichlet prior prefers sparse solutions as $\alpha \rightarrow 0$
- Gibbs sampler samples from posterior distribution of parses
  - reanalyses each word based on parses of the other words

Posterior samples from WSJ verb tokens

$\alpha = 0.1$	$\alpha = 10^{-5}$	$\alpha = 10^{-10}$	$\alpha = 10^{-15}$
expect	expect	expect	expect
expects	expects	expects	expects
expected	expected	expected	expected
expecting	expect ing	expect ing	expect ing
include	include	include	include
includes	includes	includ es	includ es
included	included	includ ed	includ ed
including	including	including	including
add	add	add	add
adds	adds	adds	add s
added	added	add ed	added
adding	adding	add ing	add ing
continue	continue	continue	continue
continues	continues	continue s	continue s
continued	continued	continu ed	continu ed
continuing	continuing	continu ing	continu ing
report	report	report	report

Relative frequencies of inflected verb forms



Types and tokens

- A word *type* is a distinct word shape
- A word *token* is an occurrence of a word

$$\begin{aligned} \text{Data} &= \text{“the cat chased the other cat”} \\ \text{Tokens} &= \text{“the”, “cat”, “chased”, “the”, “other”, “cat”} \\ \text{Types} &= \text{“the”, “cat”, “chased”, “other”} \end{aligned}$$

- Estimating θ from *word types* rather than word tokens eliminates (most) frequency variation
  - 4 common verb suffixes, so when estimating from verb types $\theta_{\text{Suffix} \rightarrow \text{i n g \#}} \approx 0.25$
- Several psycholinguists believe that humans learn morphology from word types
- Adaptor grammar mimics Goldwater et al “Interpolating between Types and Tokens” morphology-learning model

Posterior samples from WSJ verb *types*

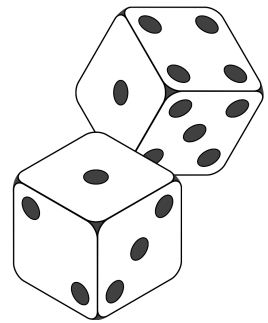
$\alpha = 0.1$		$\alpha = 10^{-5}$		$\alpha = 10^{-10}$		$\alpha = 10^{-15}$	
expect		expect		expect		exp	ect
expects		expect	s	expect	s	exp	ects
expected		expect	ed	expect	ed	exp	ected
expect	ing	expect	ing	expect	ing	exp	ecting
include		includ	e	includ	e	includ	e
include	s	includ	es	includ	es	includ	es
included		includ	ed	includ	ed	includ	ed
including		includ	ing	includ	ing	includ	ing
add		add		add		add	
adds		add	s	add	s	add	s
add	ed	add	ed	add	ed	add	ed
adding		add	ing	add	ing	add	ing
continue		continu	e	continu	e	continu	e
continue	s	continu	es	continu	es	continu	es
continu	ed	continu	ed	continu	ed	continu	ed
continuing		continu	ing	continu	ing	continu	ing
report		report		repo	rt	rep	ort

Desiderata for an extension of PCFGs

- PCFG *rules are "too small"* to be effective units of generalization
  $\Rightarrow$ generalize over groups of rules
  $\Rightarrow$ units of generalization should be chosen based on data
- *Type-based inference* mitigates over-dispersion
  $\Rightarrow$ Hierarchical Bayesian model where:
  - context-free rules generate types
  - another process replicates types to produce tokens
- *Adaptor grammars:*
  - learn *probability of entire subtrees* (how a nonterminal expands to terminals)
  - use grammatical hierarchy to define a Bayesian hierarchy, from which *type-based inference naturally emerges*

Outline

Bayesian inference for Dirichlet-multinomials



- Probability of next event with *uniform Dirichlet prior* with mass α over m outcomes and *observations* $\boldsymbol{Z}_{1:n} = (Z_1, \ldots, Z_n)$

$$\mathrm{P}(Z_{n+1} = k \mid \boldsymbol{Z}_{1:n}, \alpha) \quad \propto \quad n_k(\boldsymbol{Z}_{1:n}) + \alpha/m$$

  where $n_k(\boldsymbol{Z}_{1:n})$ is number of times k appears in $\boldsymbol{Z}_{1:n}$
- Example: Coin $(m = 2)$, $\alpha = 1$, $\boldsymbol{Z}_{1:2} = (\text{heads}, \text{heads})$
  - $\mathrm{P}(Z_3 = \text{heads} \mid \boldsymbol{Z}_{1:2}, \alpha) \propto 2.5$
  - $\mathrm{P}(Z_3 = \text{tails} \mid \boldsymbol{Z}_{1:2}, \alpha) \propto 0.5$

Dirichlet-multinomials with many outcomes

- Predictive probability:

$$\mathrm{P}(Z_{n+1} = k \mid \boldsymbol{Z}_{1:n}, \alpha) \quad \propto \quad n_k(\boldsymbol{Z}_{1:n}) + \alpha/m$$

- Suppose the number of outcomes $m \gg n$. Then:

$$\mathrm{P}(Z_{n+1} = k \mid \boldsymbol{Z}_{1:n}, \alpha) \quad \propto \quad \begin{cases} n_k(\boldsymbol{Z}_{1:n}) & \text{if } n_k(\boldsymbol{Z}_{1:n}) > 0 \\ \alpha/m & \text{if } n_k(\boldsymbol{Z}_{1:n}) = 0 \end{cases}$$

- But *most outcomes will be unobserved*, so:

$$\mathrm{P}(Z_{n+1} \notin \boldsymbol{Z}_{1:n} \mid \boldsymbol{Z}_{1:n}, \alpha) \quad \propto \quad \alpha$$

From Dirichlet-multinomials to Chinese Restaurant Processes

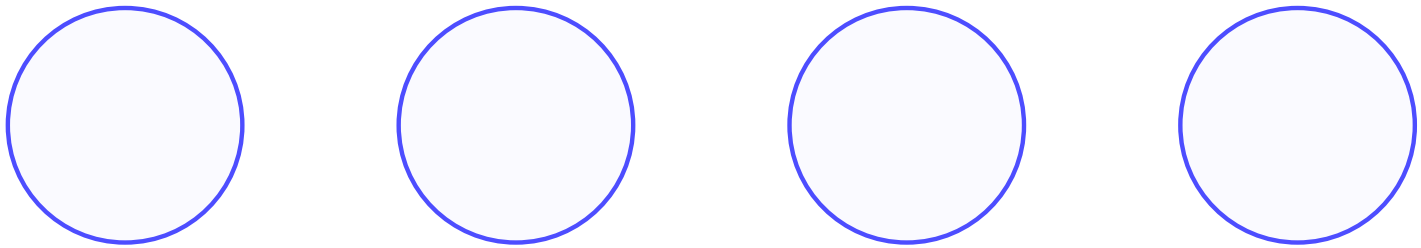

...

- Suppose *number of outcomes is unbounded* but *we* pick the event labels
- If we number event types in order of occurrence
  $\Rightarrow$ *Chinese Restaurant Process*

$$Z_1 = 1$$

$$\mathrm{P}(Z_{n+1} = k \mid \boldsymbol{Z}_{1:n}, \alpha) \propto \begin{cases} n_k(\boldsymbol{Z}_{1:n}) & \text{if } k \leq m = \max(\boldsymbol{Z}_{1:n}) \\ \alpha & \text{if } k = m + 1 \end{cases}$$

Chinese Restaurant Process (0)

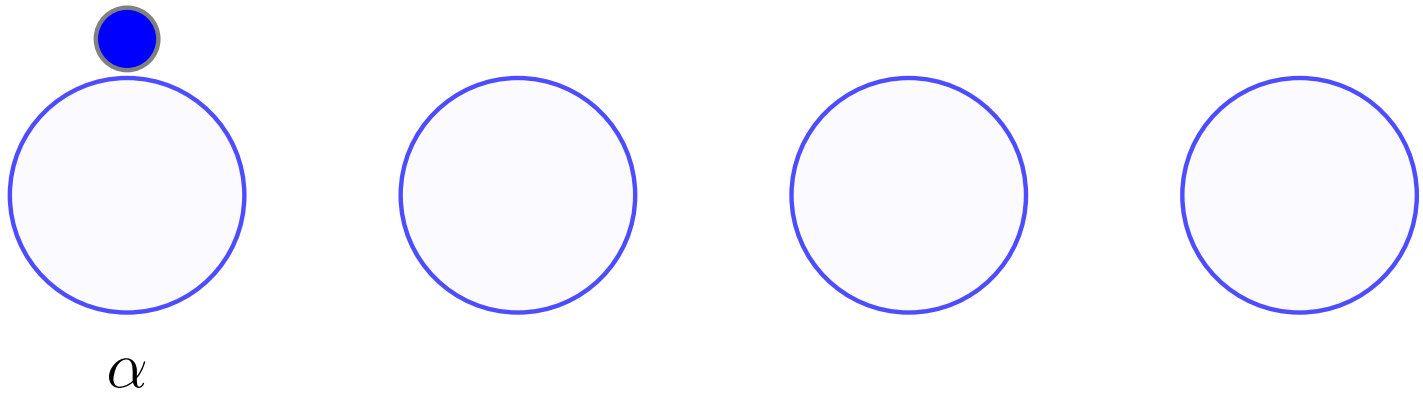


- Customer $\rightarrow$ table mapping $\boldsymbol{Z} =$
- $\mathrm{P}(\boldsymbol{z}) = 1$

- Next customer chooses a table according to:

$$\mathrm{P}(Z_{n+1} = k \mid \boldsymbol{Z}_{1:n}) \quad \propto \quad \begin{cases} n_k(\boldsymbol{Z}_{1:n}) & \text{if } k \leq m = \max(\boldsymbol{Z}_{1:n}) \\ \alpha & \text{if } k = m + 1 \end{cases}$$

Chinese Restaurant Process (1)

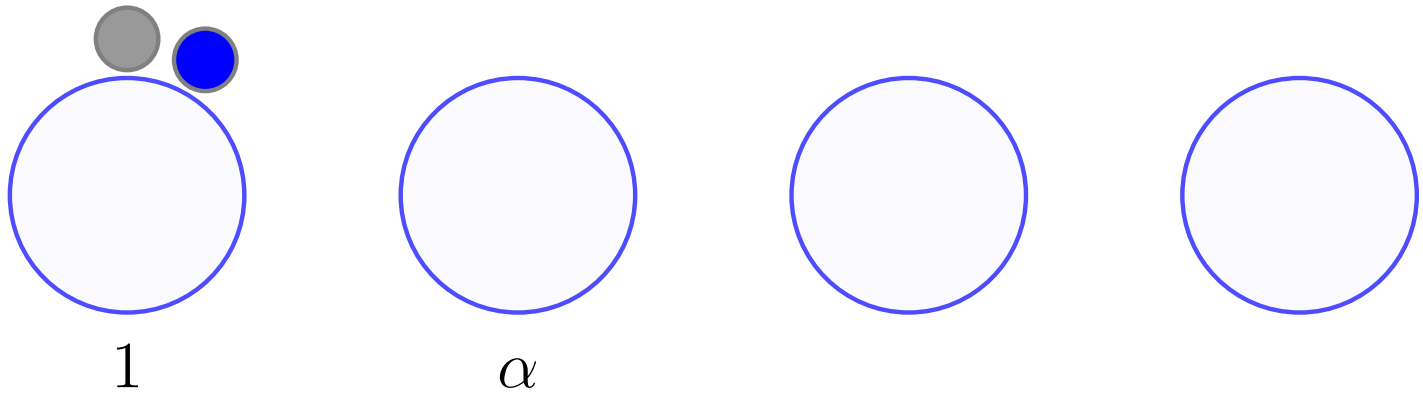


- Customer $\rightarrow$ table mapping $\boldsymbol{Z} = 1$
- $\mathrm{P}(\boldsymbol{z}) = \alpha/\alpha$

- Next customer chooses a table according to:

$$\mathrm{P}(Z_{n+1} = k \mid \boldsymbol{Z}_{1:n}) \quad \propto \quad \begin{cases} n_k(\boldsymbol{Z}_{1:n}) & \text{if } k \leq m = \max(\boldsymbol{Z}_{1:n}) \\ \alpha & \text{if } k = m + 1 \end{cases}$$

Chinese Restaurant Process (2)

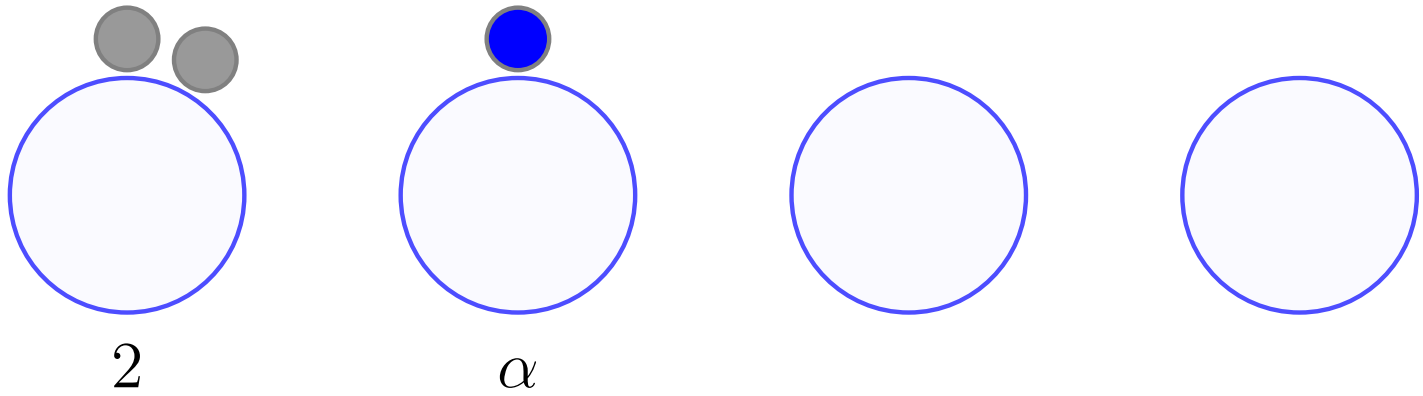


- Customer $\rightarrow$ table mapping $\boldsymbol{Z} = 1, 1$
- $\mathrm{P}(\boldsymbol{z}) = \alpha/\alpha \times 1/(1+\alpha)$

- Next customer chooses a table according to:

$$\mathrm{P}(Z_{n+1} = k \mid \boldsymbol{Z}_{1:n}) \quad \propto \quad \begin{cases} n_k(\boldsymbol{Z}_{1:n}) & \text{if } k \leq m = \max(\boldsymbol{Z}_{1:n}) \\ \alpha & \text{if } k = m+1 \end{cases}$$

Chinese Restaurant Process (3)

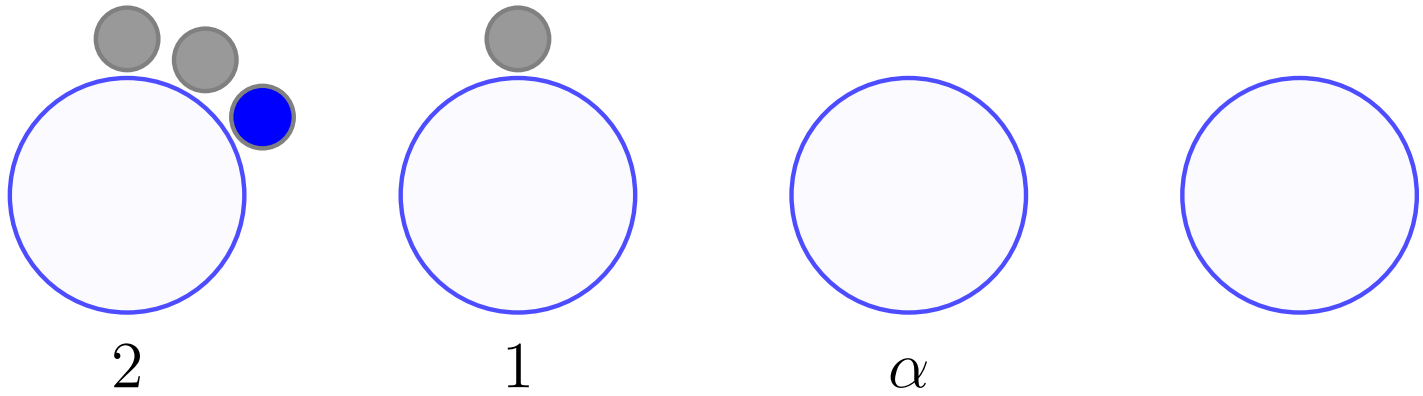


- Customer $\rightarrow$ table mapping $\boldsymbol{Z} = 1, 1, 2$
- $\mathrm{P}(\boldsymbol{z}) = \alpha/\alpha \times 1/(1+\alpha) \times \alpha/(2+\alpha)$

- Next customer chooses a table according to:

$$\mathrm{P}(Z_{n+1} = k \mid \boldsymbol{Z}_{1:n}) \quad \propto \quad \begin{cases} n_k(\boldsymbol{Z}_{1:n}) & \text{if } k \leq m = \max(\boldsymbol{Z}_{1:n}) \\ \alpha & \text{if } k = m+1 \end{cases}$$

Chinese Restaurant Process (4)

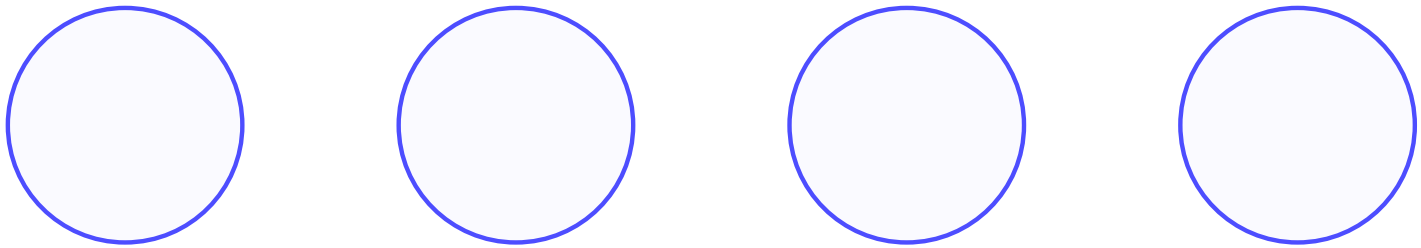


- Customer $\rightarrow$ table mapping $\boldsymbol{Z} = 1, 1, 2, 1$
- $\mathrm{P}(\boldsymbol{z}) = \alpha/\alpha \times 1/(1+\alpha) \times \alpha/(2+\alpha) \times 2/(3+\alpha)$

- Next customer chooses a table according to:

$$\mathrm{P}(Z_{n+1} = k \mid \boldsymbol{Z}_{1:n}) \quad \propto \quad \begin{cases} n_k(\boldsymbol{Z}_{1:n}) & \text{if } k \leq m = \max(\boldsymbol{Z}_{1:n}) \\ \alpha & \text{if } k = m+1 \end{cases}$$

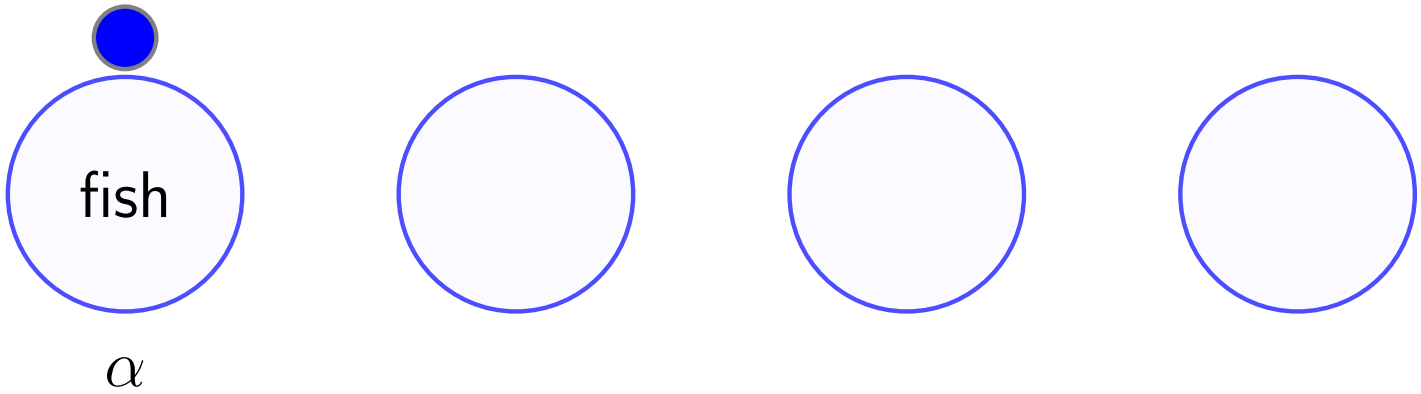
Labeled Chinese Restaurant Process (0)



- Table $\rightarrow$ label mapping $\boldsymbol{Y} =$
- Customer $\rightarrow$ table mapping $\boldsymbol{Z} =$
- Output sequence $\boldsymbol{X} =$
- $\mathrm{P}(\boldsymbol{X}) = 1$

- *Base distribution* $\mathrm{P}_0(Y)$ generates a *label* Y_k for each table k
- All customers sitting at table k (i.e., $Z_i = k$) share label Y_k
- Customer i sitting at table Z_i has label $X_i = Y_{Z_i}$

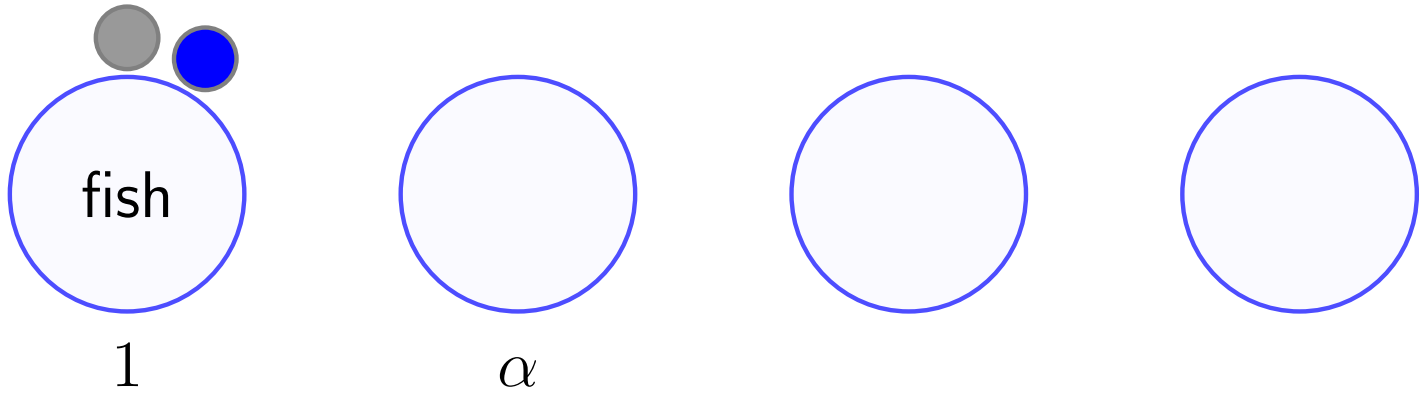
Labeled Chinese Restaurant Process (1)



- Table $\rightarrow$ label mapping $\boldsymbol{Y} =$ fish
- Customer $\rightarrow$ table mapping $\boldsymbol{Z} = 1$
- Output sequence $\boldsymbol{X} =$ fish
- $\mathrm{P}(\boldsymbol{X}) = \alpha/\alpha \times \mathrm{P}_0(\text{fish})$

- *Base distribution* $\mathrm{P}_0(Y)$ generates a *label* Y_k for each table k
- All customers sitting at table k (i.e., $Z_i = k$) share label Y_k
- Customer i sitting at table Z_i has label $X_i = Y_{Z_i}$

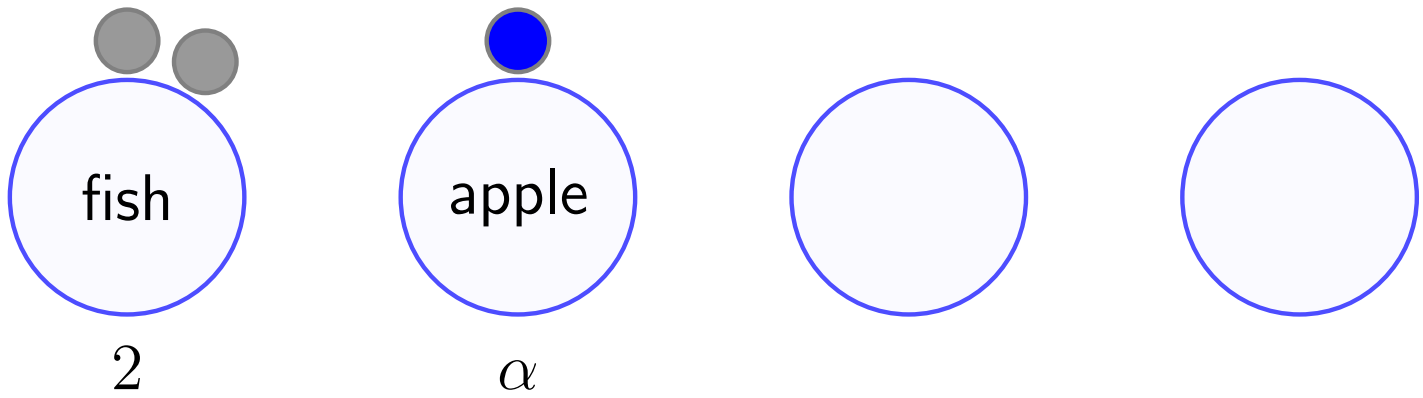
Labeled Chinese Restaurant Process (2)



- Table $\rightarrow$ label mapping $\boldsymbol{Y} =$ fish
- Customer $\rightarrow$ table mapping $\boldsymbol{Z} = 1, 1$
- Output sequence $\boldsymbol{X} =$ fish,fish
- $\mathrm{P}(\boldsymbol{X}) = \mathrm{P}_0(\text{fish}) \times 1/(1+\alpha)$

- *Base distribution* $\mathrm{P}_0(Y)$ generates a *label* Y_k for each table k
- All customers sitting at table k (i.e., $Z_i = k$) share label Y_k
- Customer i sitting at table Z_i has label $X_i = Y_{Z_i}$

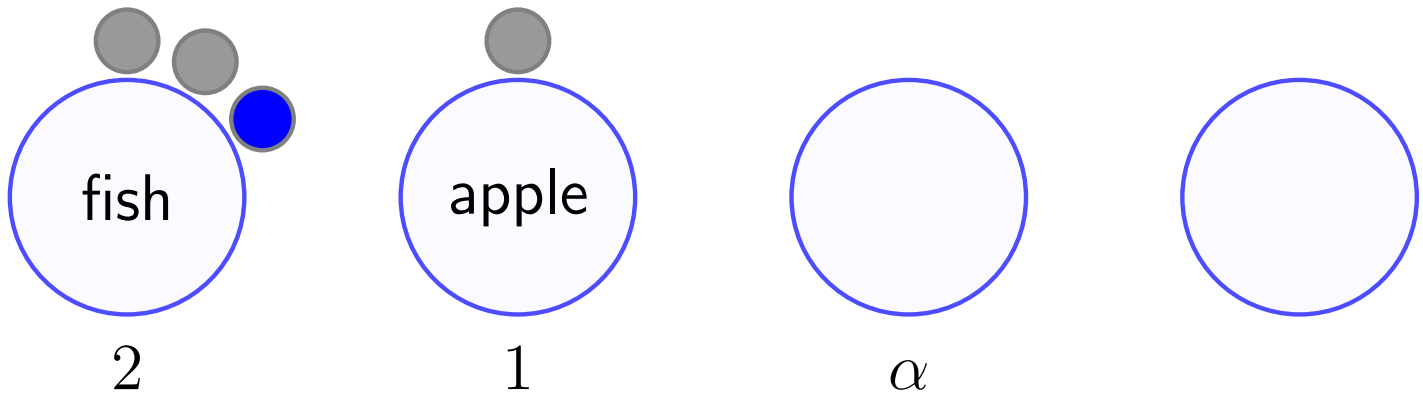
Labeled Chinese Restaurant Process (3)



- Table $\rightarrow$ label mapping $\boldsymbol{Y} =$ fish,apple
- Customer $\rightarrow$ table mapping $\boldsymbol{Z} = 1, 1, 2$
- Output sequence $\boldsymbol{X} =$ fish,fish,apple
- $\mathrm{P}(\boldsymbol{X}) = \mathrm{P}_0(\text{fish}) \times 1/(1+\alpha) \times \alpha/(2+\alpha)\mathrm{P}_0(\text{apple})$

- *Base distribution* $\mathrm{P}_0(Y)$ generates a *label* Y_k for each table k
- All customers sitting at table k (i.e., $Z_i = k$) share label Y_k
- Customer i sitting at table Z_i has label $X_i = Y_{Z_i}$

Labeled Chinese Restaurant Process (4)



- Table $\rightarrow$ label mapping $\boldsymbol{Y} =$ fish,apple
- Customer $\rightarrow$ table mapping $\boldsymbol{Z} = 1, 1, 2$
- Output sequence $\boldsymbol{X} =$ fish,fish,apple,fish
- $\mathrm{P}(\boldsymbol{X}) = \mathrm{P}_0(\text{fish}) \times 1/(1+\alpha) \times \alpha/(2+\alpha)\mathrm{P}_0(\text{apple}) \times 2/(3+\alpha)$

- *Base distribution* $\mathrm{P}_0(Y)$ generates a *label* Y_k for each table k
- All customers sitting at table k (i.e., $Z_i = k$) share label Y_k
- Customer i sitting at table Z_i has label $X_i = Y_{Z_i}$

Summary: Chinese Restaurant Processes

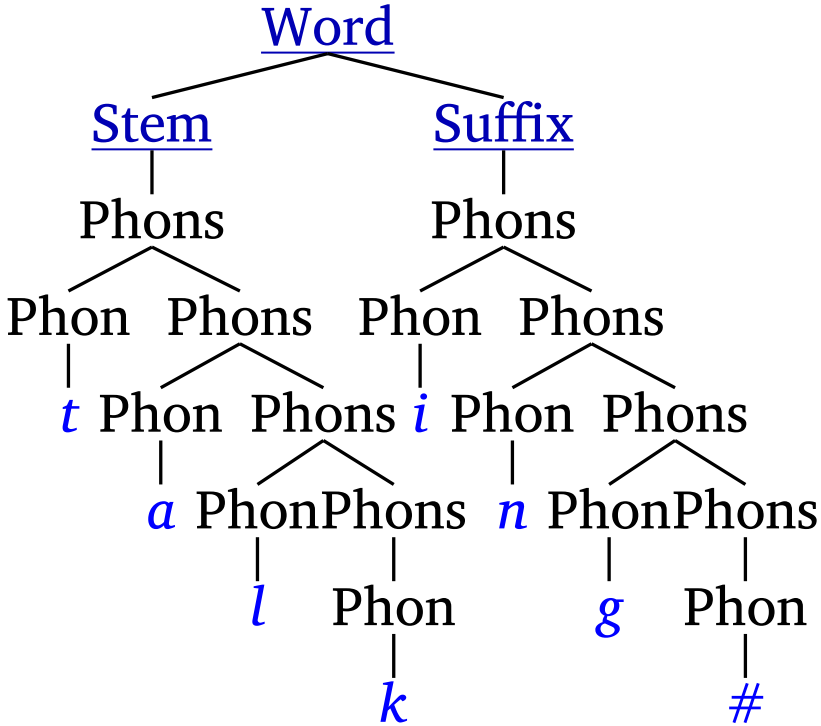
- *Chinese Restaurant Processes* (CRPs) generalise Dirichlet-Multinomials to an *unbounded number of outcomes*
  - *concentration parameter* α controls how likely a new outcome is
  - CRPs exhibit a *rich get richer* power-law behaviour
- *Pitman-Yor Processes* (PYPs) generalise CRPs with an additional concentration parameter
  - this parameter specifies the asymptotic power-law behaviour
- *Labeled CRPs* use a *base distribution* to define distributions over arbitrary objects
  - base distribution *"labels the tables"*
  - base distribution can have *infinite support*
  - concentrates mass on a countable subset
  - power-law behaviour $\Rightarrow$ Zipfian distributions

Outline

Adaptor grammars: informal description

- The trees generated by an adaptor grammar are defined by CFG rules as in a CFG
- A subset of the nonterminals are *adapted*
- *Unadapted nonterminals* expand by picking a rule and recursively expanding its children, as in a PCFG
- *Adapted nonterminals* can expand in two ways:
  - by picking a rule and recursively expanding its children, or
  - by *generating a previously generated tree, with probability proportional to the number of times previously generated*

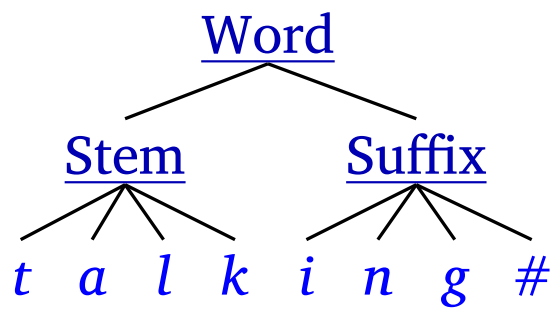
Adaptor grammar for stem-suffix morphology



Word → Stem Suffix
Stem → Phons
Suffix → Phons
Phons → Phon
Phons → Phon Phons

or in *abbreviated form* with
non-adapted nonterminals suppressed

Word → Stem Suffix
Stem → Phon^+
Suffix → Phon^+



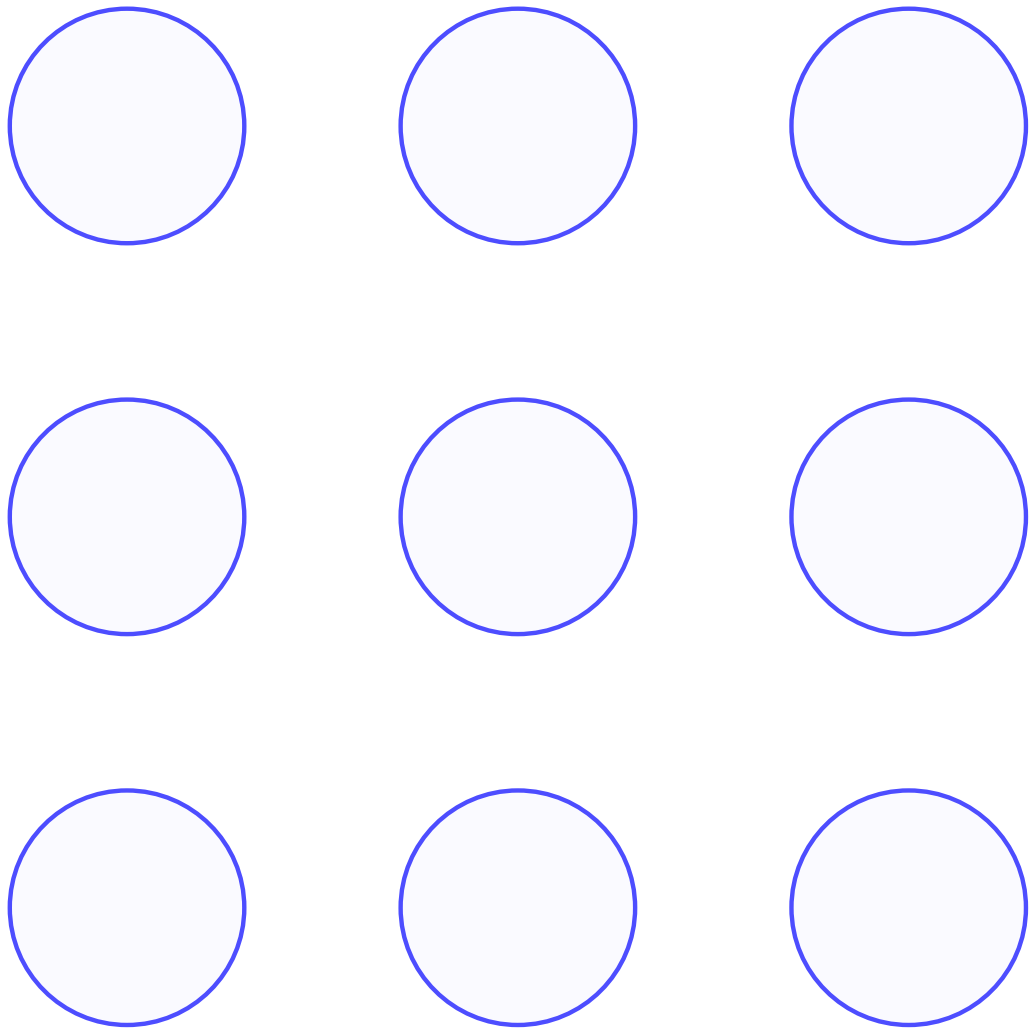
MACQUARIE UNIVERSITY

Adaptor grammar for stem-suffix morphology (0)

Word $\rightarrow$ Stem Suffix

Stem $\rightarrow$ Phoneme^{+}

Suffix $\rightarrow$ Phoneme*



Generated words:

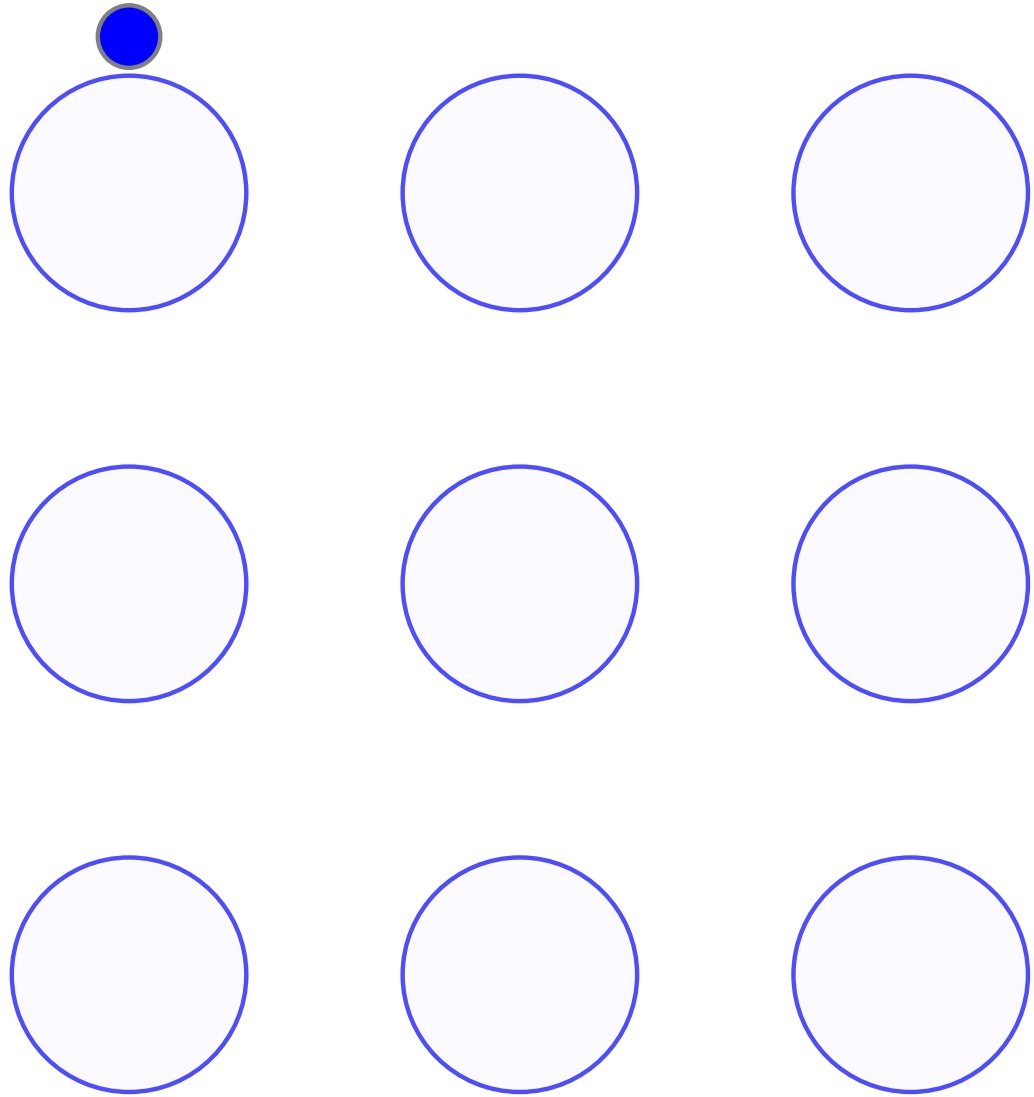
MACQUARIE UNIVERSITY

Adaptor grammar for stem-suffix morphology (1a)

Word $\rightarrow$ Stem Suffix

Stem $\rightarrow$ Phoneme^{+}

Suffix $\rightarrow$ Phoneme*



Generated words:

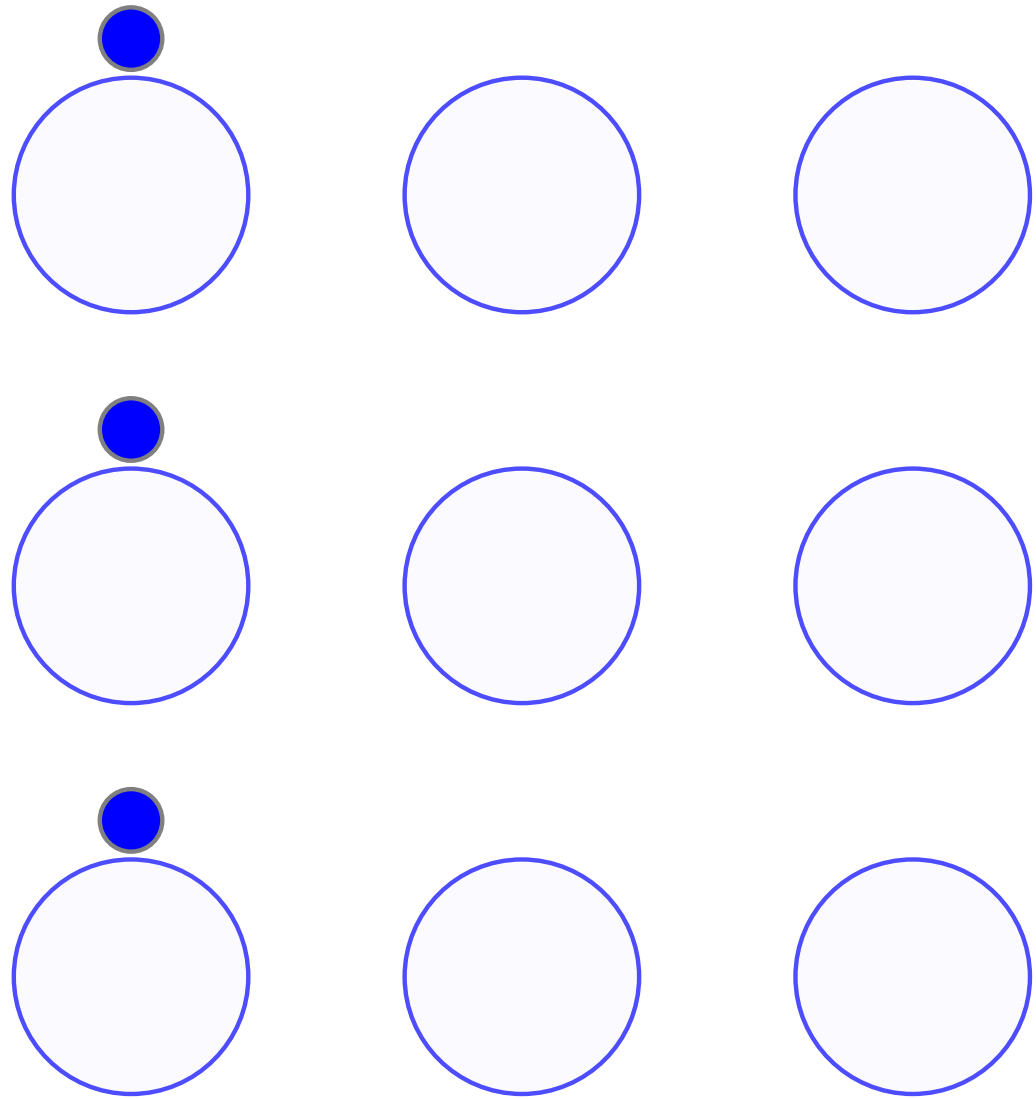
MACQUARIE UNIVERSITY

Adaptor grammar for stem-suffix morphology (1b)

Word $\rightarrow$ Stem Suffix

Stem $\rightarrow$ Phoneme^{+}

Suffix $\rightarrow$ Phoneme*



Generated words:

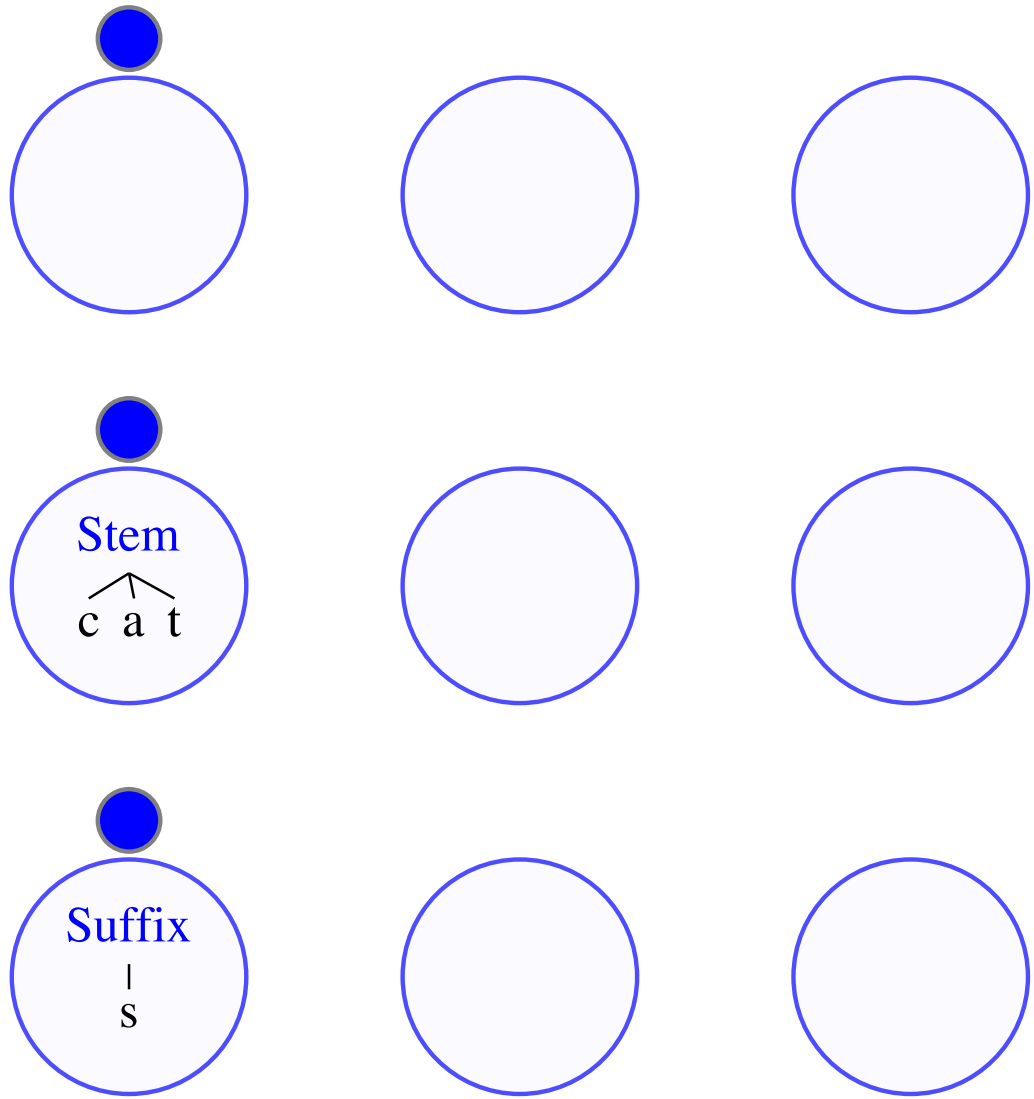
MACQUARIE UNIVERSITY

Adaptor grammar for stem-suffix morphology (1c)

Word $\rightarrow$ Stem Suffix

Stem $\rightarrow$ Phoneme^{+}

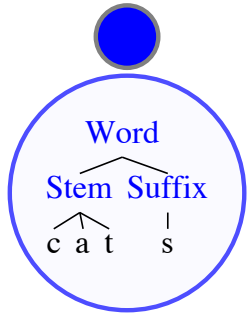
Suffix $\rightarrow$ Phoneme*



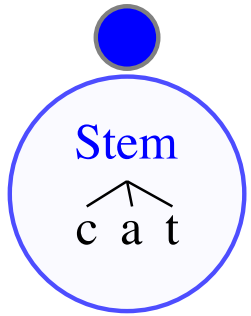
Generated words:

Adaptor grammar for stem-suffix morphology (1d)

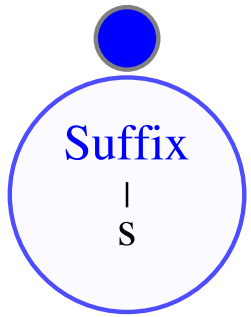
Word → Stem Suffix



Stem → Phoneme^{+}



Suffix → Phoneme*

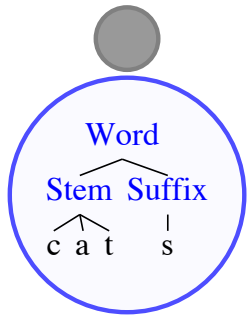


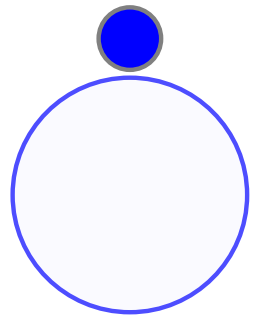
Generated words: cats

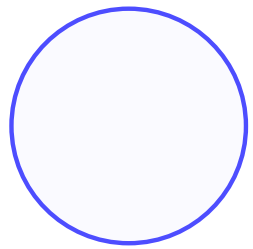
MACQUARIE UNIVERSITY

Adaptor grammar for stem-suffix morphology (2a)

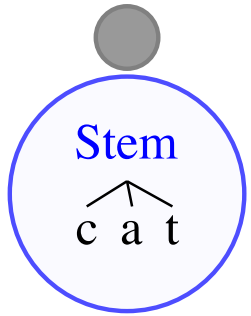
Word → Stem Suffix

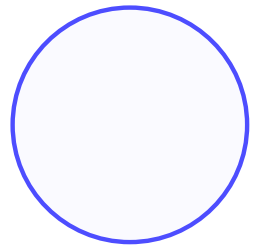


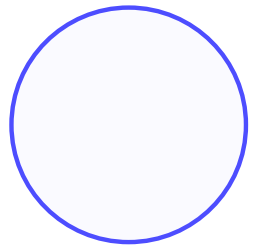




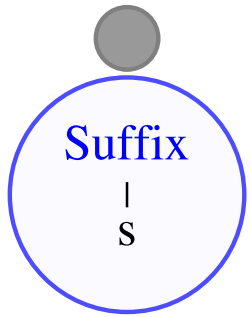
Stem → Phoneme^{+}

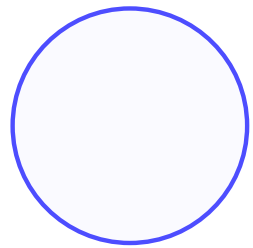


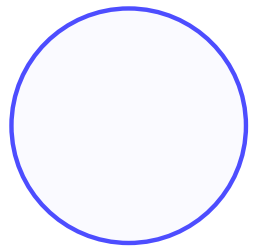




Suffix → Phoneme*



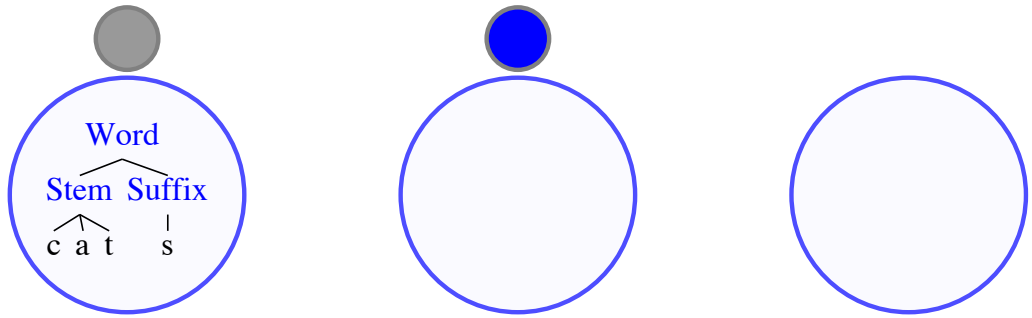




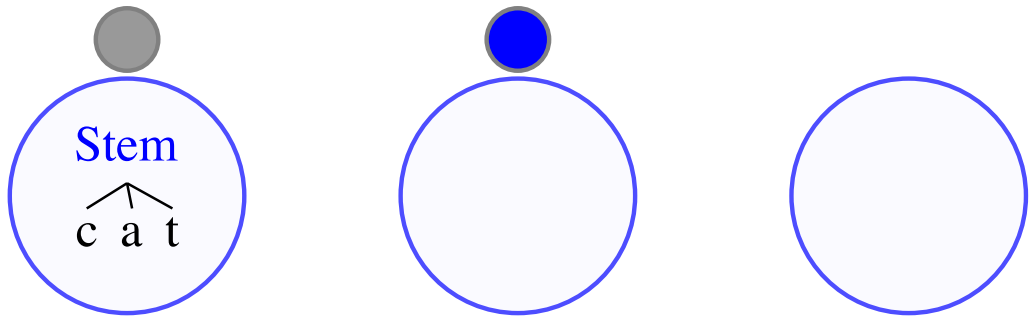
Generated words: cats

Adaptor grammar for stem-suffix morphology (2b)

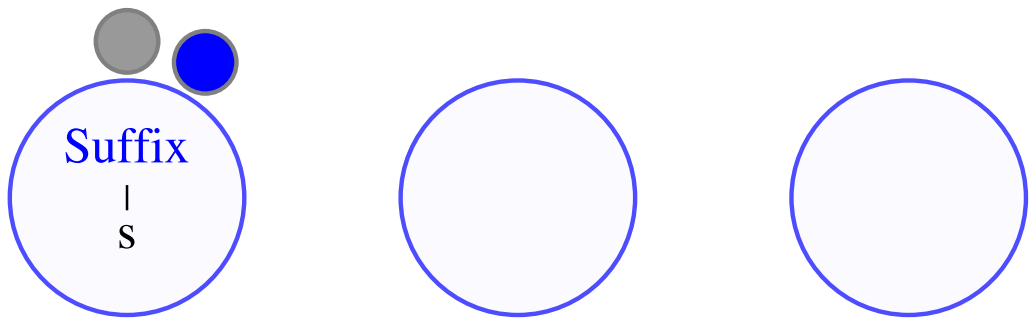
Word → Stem Suffix



Stem → Phoneme^{+}



Suffix → Phoneme*



Generated words: cats

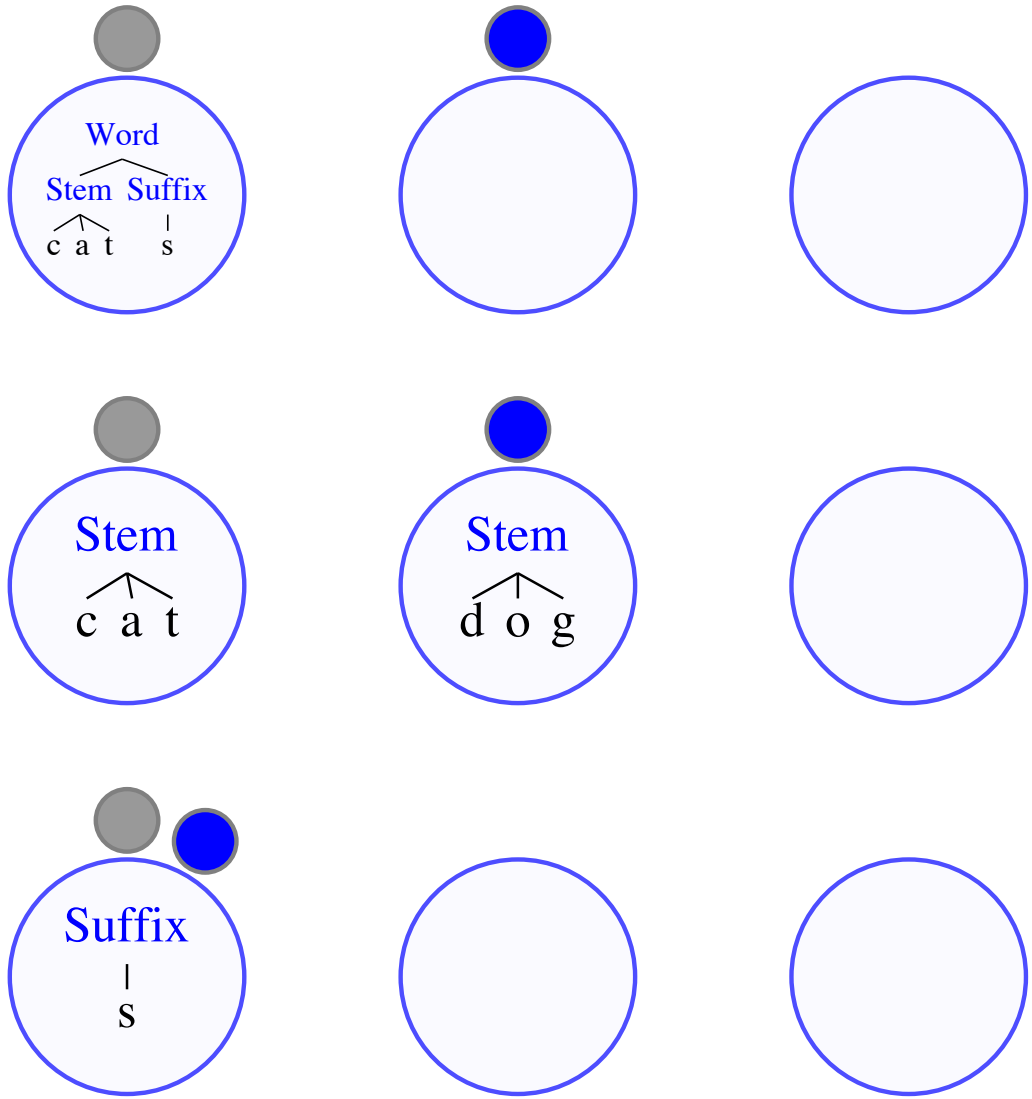
MACQUARIE UNIVERSITY

Adaptor grammar for stem-suffix morphology (2c)

Word → Stem Suffix

Stem → Phoneme^{+}

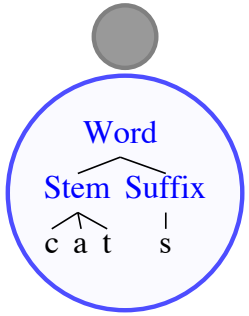
Suffix → Phoneme*

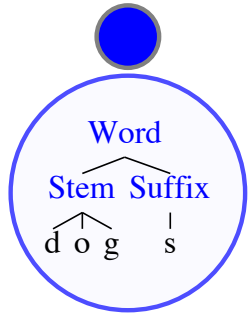


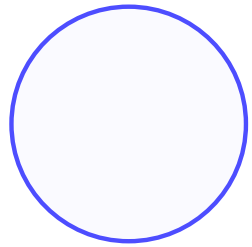
Generated words: cats

Adaptor grammar for stem-suffix morphology (2d)

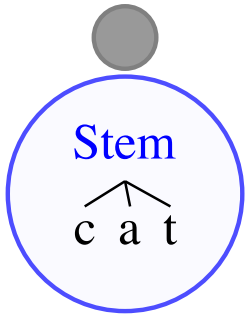
Word → Stem Suffix

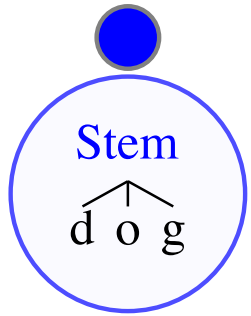


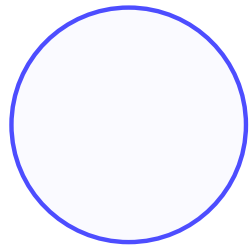




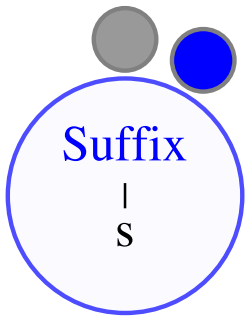
Stem → Phoneme^{+}

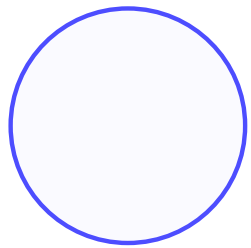


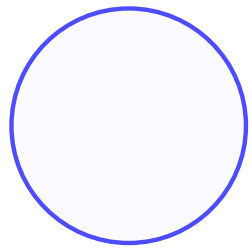




Suffix → Phoneme*





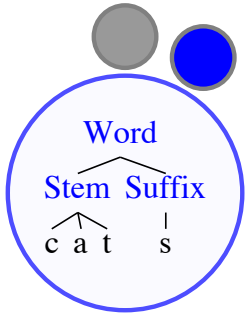


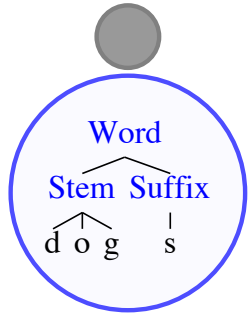
Generated words: cats, dogs

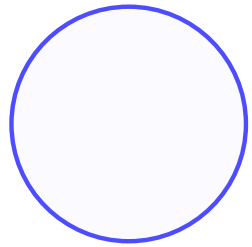
MACQUARIE UNIVERSITY

Adaptor grammar for stem-suffix morphology (3)

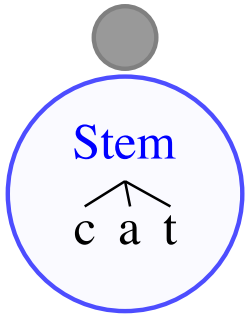
Word → Stem Suffix

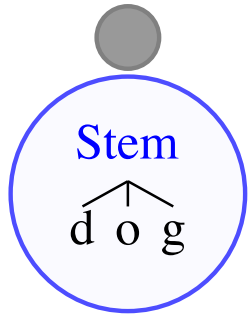


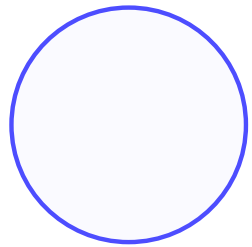




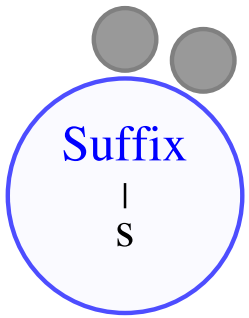
Stem → Phoneme^{+}

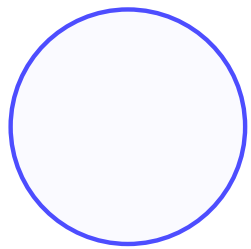


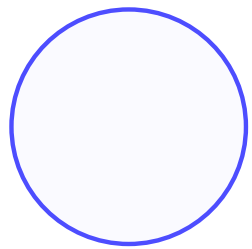




Suffix → Phoneme*







Generated words: cats, dogs, cats

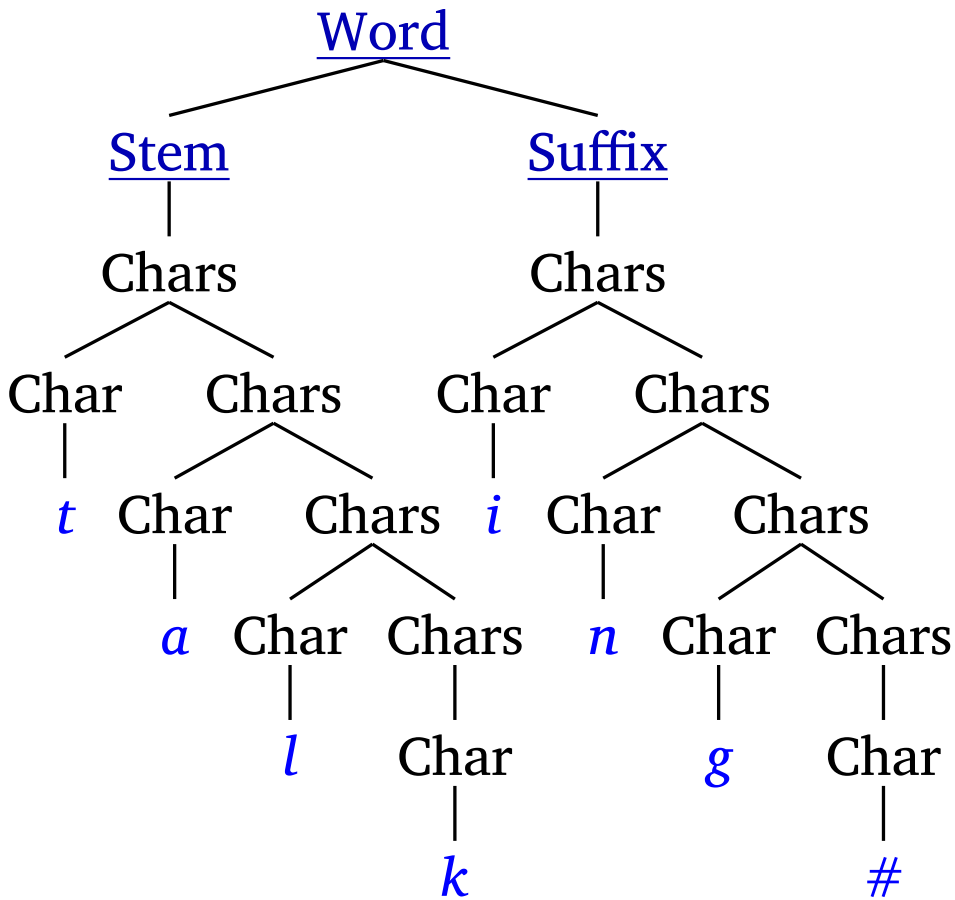
MACQUARIE UNIVERSITY

Properties of adaptor grammars

- Probability of reusing an adapted subtree T_A
  $\propto$ number of times T_A was previously generated
  - adapted subtrees are *not independent*
    - an adapted subtree can be *more probable* than the rules used to construct it
  - but they are *exchangable* $\Rightarrow$ efficient sampling algorithms
  - "rich get richer" $\Rightarrow$ Zipf power-law distributions
- Each adapted nonterminal is associated with a *Chinese Restaurant Process* or *Pitman-Yor Process*
  - CFG rules define *base distribution* of CRP or PYP
- CRP/PYP parameters (e.g., α_A) can themselves be estimated (e.g., slice sampling), so *AG models have no tunable parameters*

Bayesian hierarchy inverts grammatical hierarchy

- Grammatically, a Word is composed of a Stem and a Suffix, which are composed of Chars
- To generate a new Word from an Adaptor Grammar:
  - reuse an old Word, or
  - generate a fresh one from the base distribution, i.e., generate a Stem and a Suffix



- *Lower in the tree ⇒ higher in Bayesian hierarchy*

Learning an adaptor grammar

- *Sampling algorithms* are a natural implementation because they sample the parameters (tree fragments) as well as their values
  - *sufficient statistics* are *number of times each rule and tree fragment* is used in a derivation
- Gibbs sampler for learning an AG from strings alone:
  1. select a training sentence (e.g., at random)
  2. (subtract sentence's rule and fragment counts if parsed before)
  3. *sample a parse for sentence* given parses of other sentences
     - *PCFG proposal distribution* and MH correction
  4. add new parse's rule and tree fragment counts to global totals
  - repeat steps 1.–4. forever
- Computationally-intensive step is *parsing sentences of training data*

Outline

MACQUARIE UNIVERSITY

Unsupervised word segmentation

- Input: phoneme sequences with *sentence boundaries* (Brent)
- Task: identify *word boundaries*, and hence words

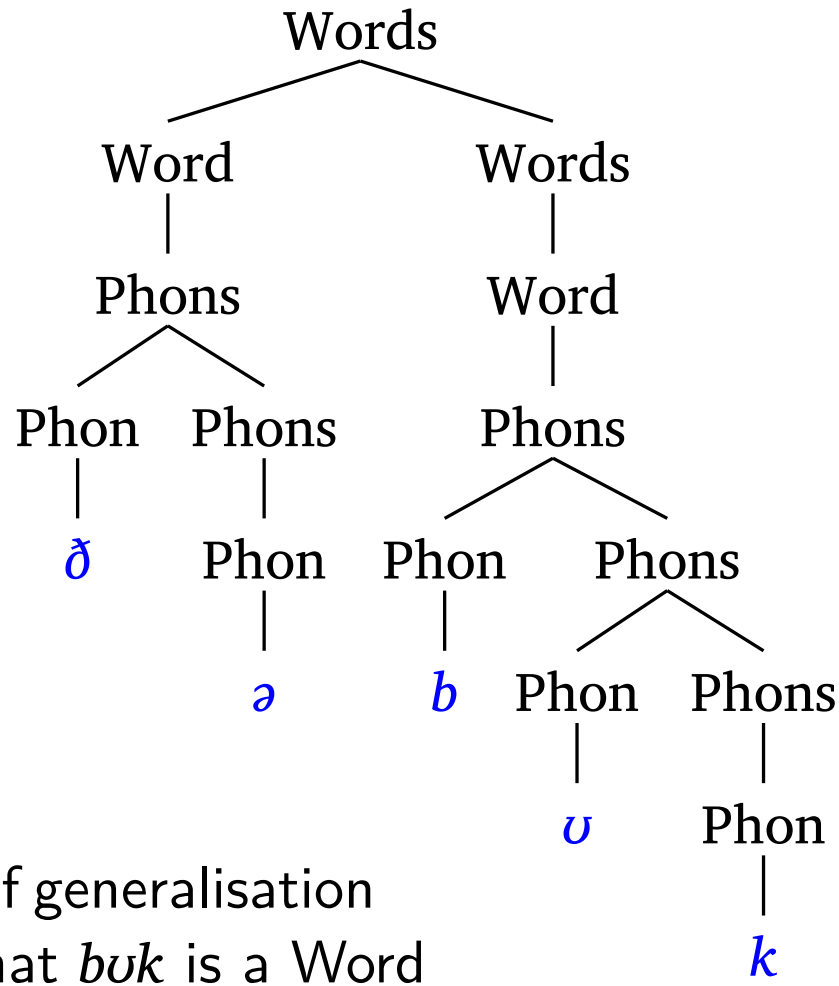
$$j \triangle u \blacktriangle w \triangle a \triangle n \triangle t \blacktriangle t \triangle u \blacktriangle s \triangle i \blacktriangle ð \triangle ə \blacktriangle b \triangle ʊ \triangle k$$

"you want to see the book"

- Ignoring phonology and morphology, this involves learning the pronunciations of the lexical items in the language

PCFG model of word segmentation

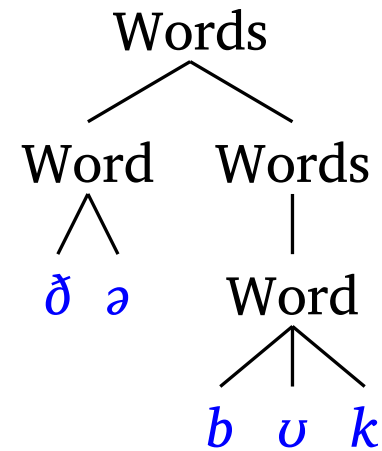
Words → Word
Words → Word Words
Word → Phons
Phons → Phon
Phons → Phon Phons
Phon → *a* | *b* | ...



- CFG trees can *describe* segmentation, but
- PCFGs *can't distinguish* good segmentations from bad ones
  - PCFG rules are *too small* a unit of generalisation
  - need to learn e.g., probability that *bʊk* is a Word

Towards non-parametric grammars

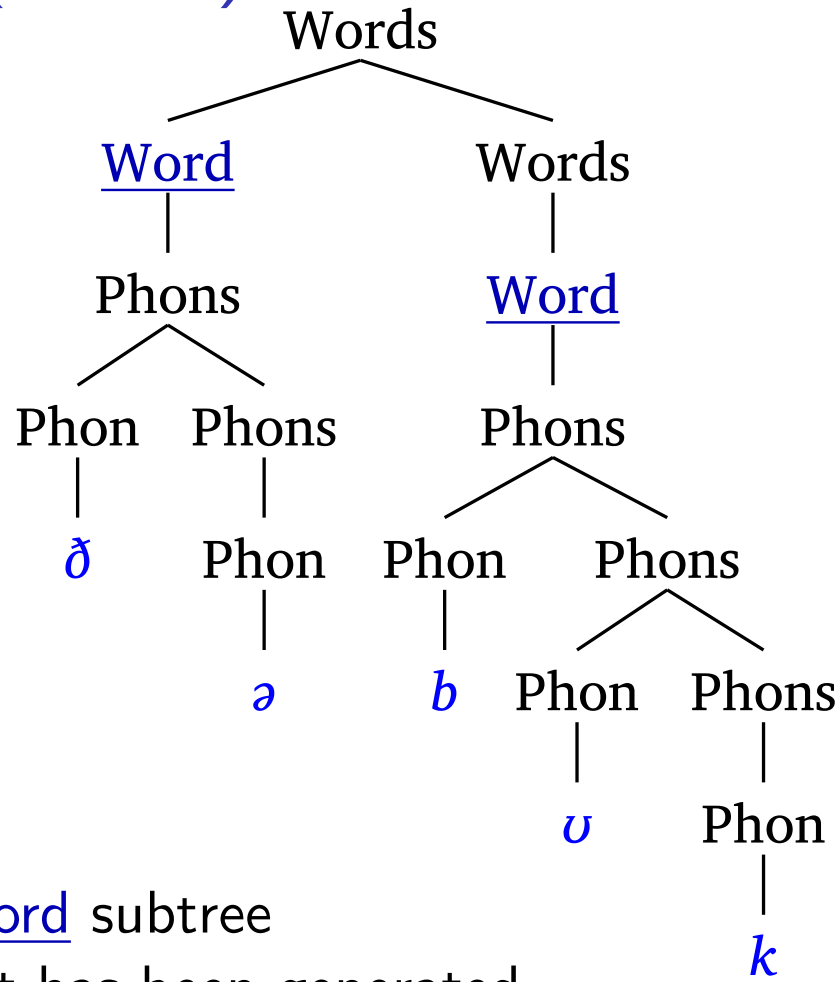
Words → Word
Words → Word Words
Word → *all possible phoneme sequences*



- Learn probability Word → $b\ \upsilon\ k$
- But *infinitely many possible Word expansions*
  ⇒ this grammar is *not a PCFG*
- Given *fixed training data*, only finitely many useful rules
  ⇒ use data to choose Word rules as well as their probabilities
- An adaptor grammar can do precisely this!

Unigram adaptor grammar (Brent)

Words → Word
Words → Word Words
Word → Phons
Phons → Phon
Phons → Phon Phons



- Word nonterminal is adapted

⇒ To generate a Word:

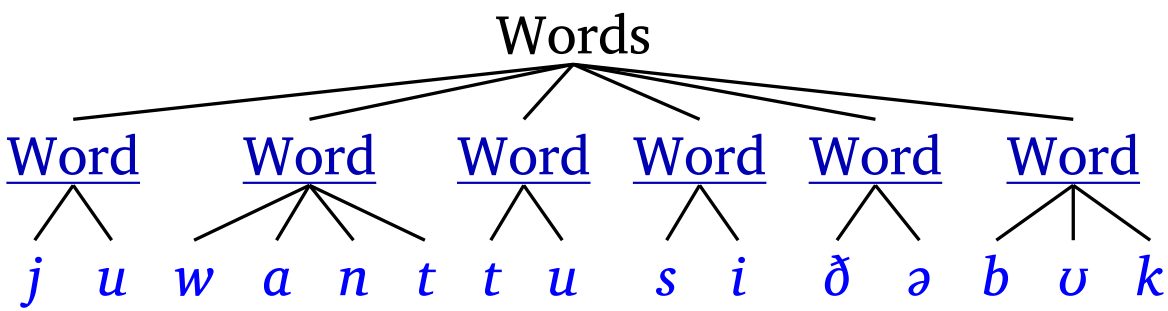
  - select a previously generated Word subtree with prob. $\propto$ number of times it has been generated
  - expand using Word → Phons rule with prob. $\propto \alpha_{\text{Word}}$ and recursively expand Phons

Unigram model of word segmentation

- Unigram "bag of words" model (Brent):
  - generate a *dictionary*, i.e., a set of words, where each word is a random sequence of phonemes
    - Bayesian prior prefers smaller dictionaries
  - generate each utterance by choosing each word at random from dictionary
- Brent's unigram model as an adaptor grammar:

$\text{Words} \rightarrow \text{Word}^+$

$\underline{\text{Word}} \rightarrow \text{Phoneme}^+$



- Accuracy of word segmentation learnt: *56% token f-score* (same as Brent model)
- But we can construct many more word segmentation models using AGs

Adaptor grammar learnt from Brent corpus

- **Initial grammar**

1	Words → Word Words	1	Words → Word
1	Word → Phon		
1	Phons → Phon Phons	1	Phons → Phon
1	Phon → *D*	1	Phon → *G*
1	Phon → *A*	1	Phon → *E*

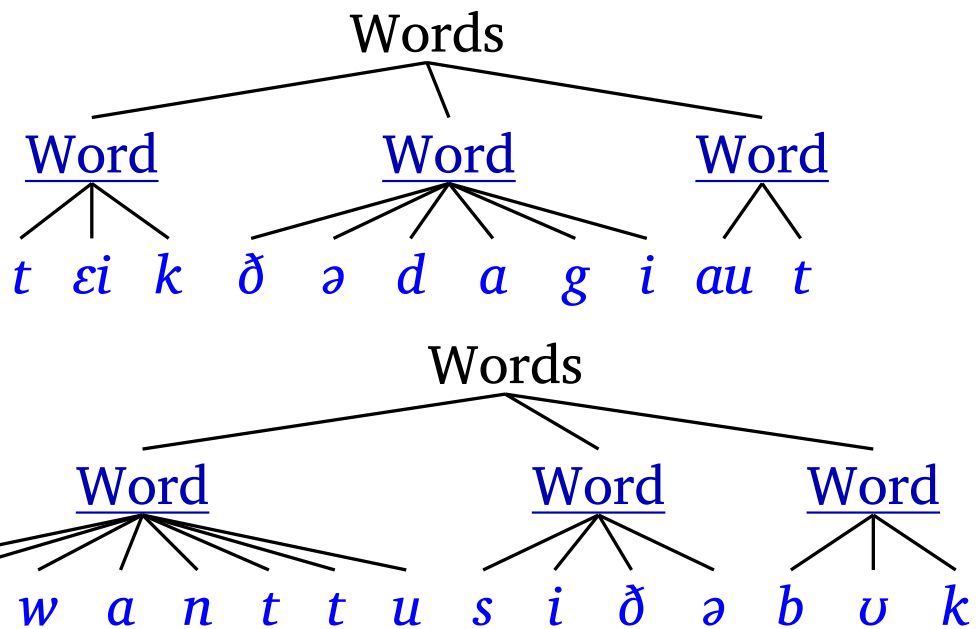
- **A grammar learnt from Brent corpus**

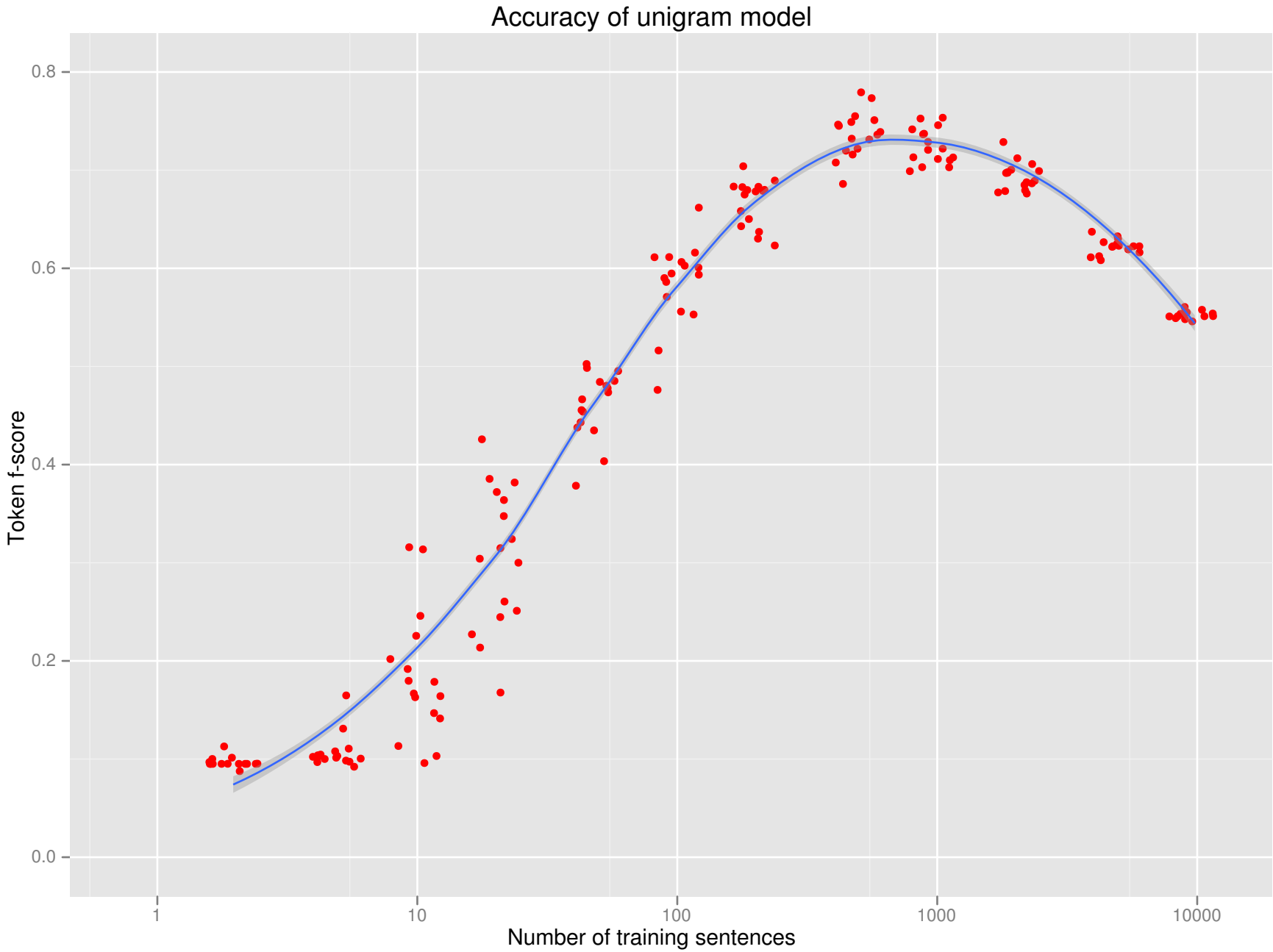
16625	Words → Word Words	9791	Words → Word
1575	Word → Phons		
4962	Phons → Phon Phons	1575	Phons → Phon
134	Phon → *D*	41	Phon → *G*
180	Phon → *A*	152	Phon → *E*
460	Word → (Phons (Phon *y*) (Phons (Phon *u*)))		
446	Word → (Phons (Phon *w*) (Phons (Phon *A*) (Phons (Phon *t*))))		
374	Word → (Phons (Phon *D*) (Phons (Phon *6*)))		
372	Word → (Phons (Phon *&*) (Phons (Phon *n*) (Phons (Phon *d*))))		

Undersegmentation errors with Unigram model

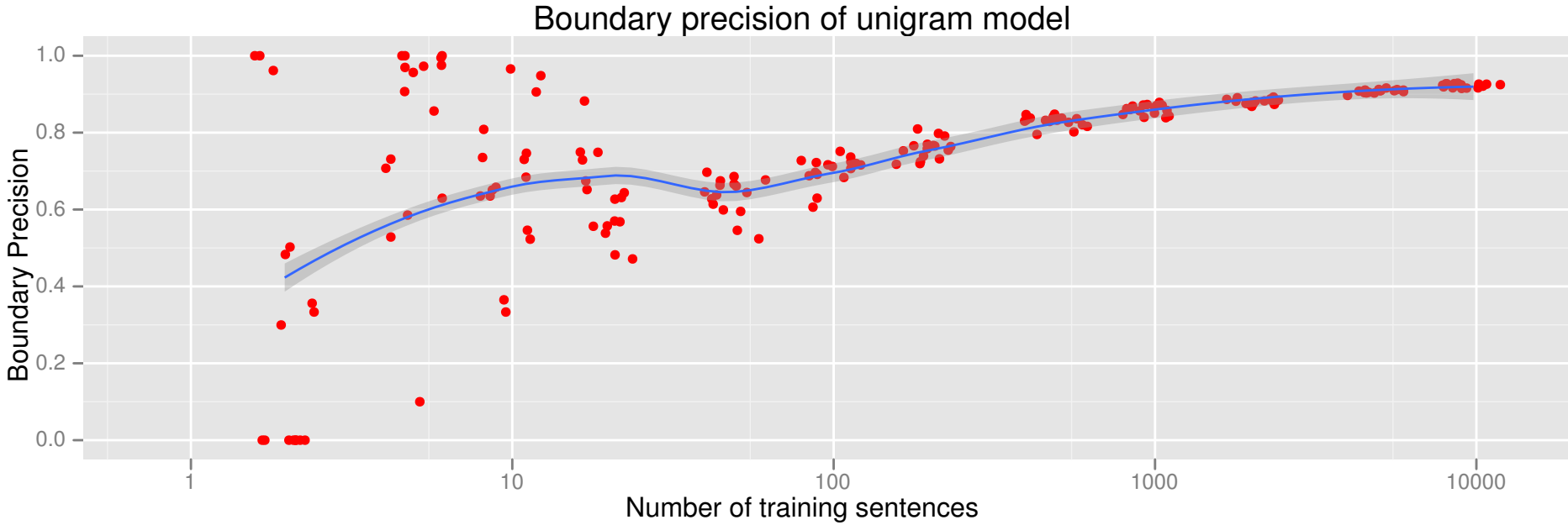
$$\text{Words} \rightarrow \underline{\text{Word}}^{+} \qquad \underline{\text{Word}} \rightarrow \text{Phon}^{+}$$

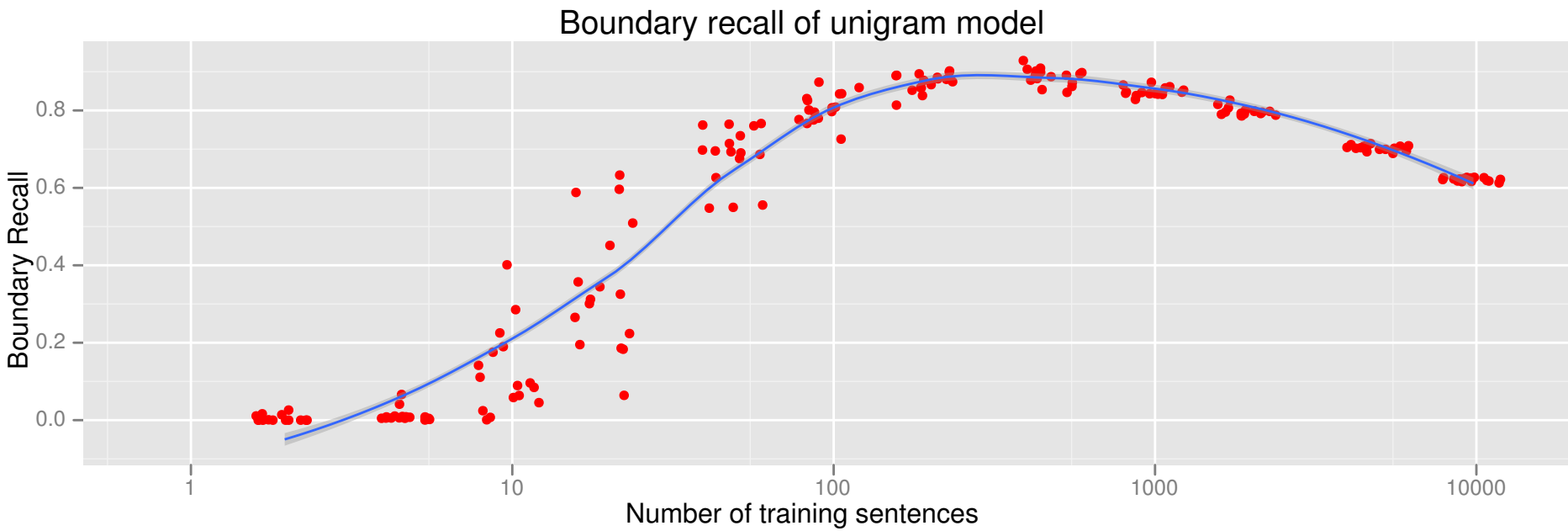
- Unigram word segmentation model assumes each word is generated independently
- But there are strong inter-word dependencies (collocations)
- Unigram model can only capture such dependencies by analyzing collocations as words (Goldwater 2006)





MACQUARIE UNIVERSITY

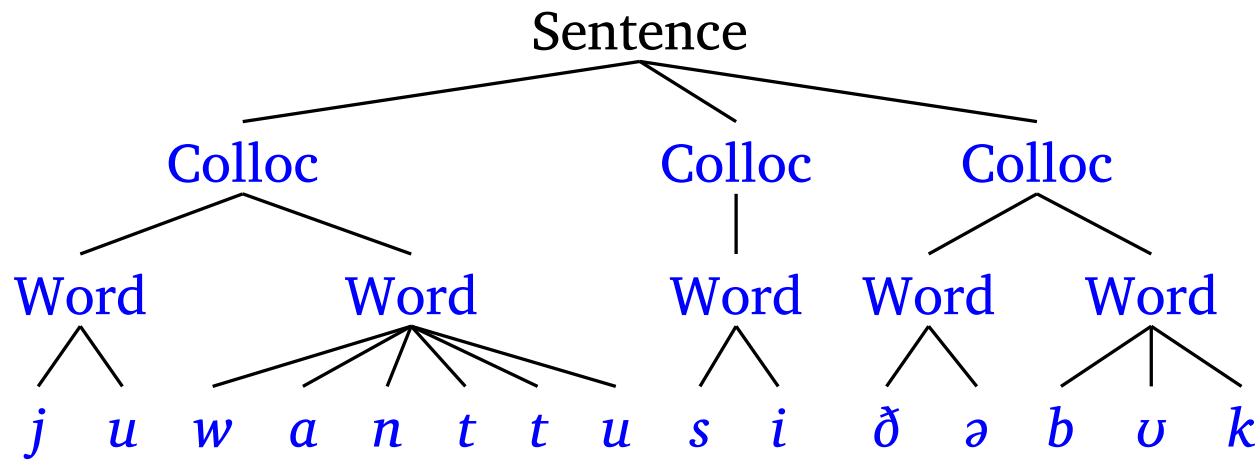




MACQUARIE UNIVERSITY

Collocations ⇒ Words

$$\text{Sentence} \rightarrow \text{Colloc}^+$$
$$\underline{\text{Colloc}} \rightarrow \text{Word}^+$$
$$\underline{\text{Word}} \rightarrow \text{Phon}^+$$



- A Colloc(ation) consists of one or more words
- Both Words and Collocs are adapted (learnt)
- Significantly improves word segmentation accuracy over unigram model (76% f-score; ≈ Goldwater's bigram model)

Outline

Two hypotheses about language acquisition

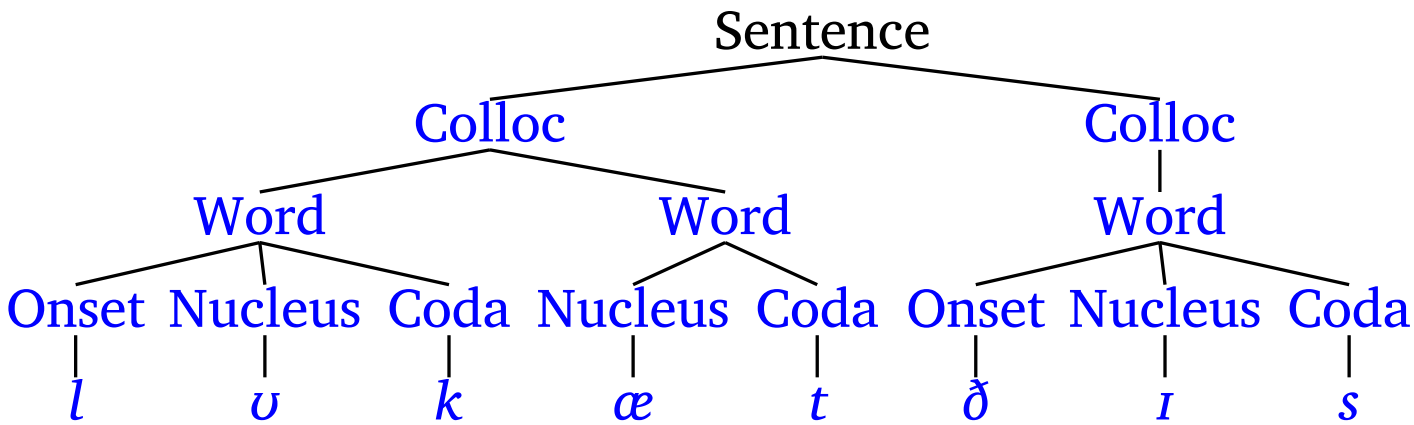
1. Pre-programmed *staged acquisition* of linguistic components
   - Conventional view of *lexical acquisition*, e.g., Kuhl (2004)
     - child first learns the phoneme inventory, which it then uses to learn
     - phonotactic cues for word segmentation, which are used to learn
     - phonological forms of words in the lexicon, . . .
2. *Interactive acquisition* of all linguistic components together
   - corresponds to *joint inference* for all components of language
   - stages in language acquisition might be due to:
     - child's input may contain more information about some components
     - some components of language may be learnable with less data

Synergies: an advantage of interactive learning

- An *interactive learner* can take advantage of *synergies in acquisition*
  - partial knowledge of component A provides information about component B
  - partial knowledge of component B provides information about component A
- A staged learner can only take advantage of one of these dependencies
- An interactive or *joint learner* can benefit from a positive feedback cycle between A and B
- Are there synergies in *learning how to segment words* and *identifying the referents of words*?

Jointly learning words and syllables

$$\text{Sentence} \rightarrow \text{Colloc}^{+} \qquad \underline{\text{Colloc}} \rightarrow \text{Word}^{+}$$
$$\underline{\text{Word}} \rightarrow \text{Syllable}^{\{1:3\}} \qquad \text{Syllable} \rightarrow (\text{Onset})\ \text{Rhyme}$$
$$\underline{\text{Onset}} \rightarrow \text{Consonant}^{+} \qquad \text{Rhyme} \rightarrow \text{Nucleus}\ (\text{Coda})$$
$$\underline{\text{Nucleus}} \rightarrow \text{Vowel}^{+} \qquad \underline{\text{Coda}} \rightarrow \text{Consonant}^{+}$$

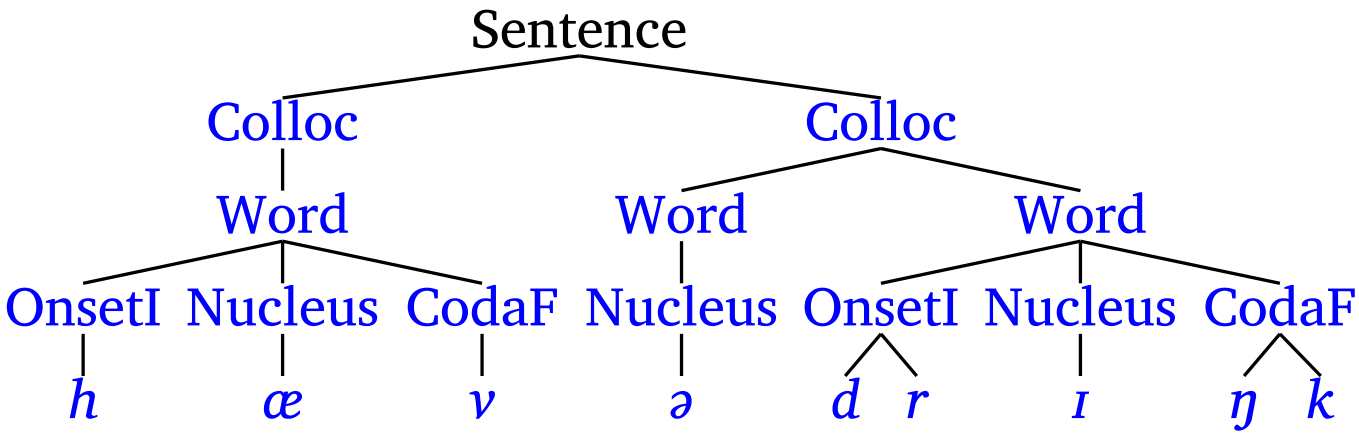


- Rudimentary syllable model (an improved model might do better)
- With 2 Collocation levels, f-score = 84%

Distinguishing internal onsets/codas helps

Sentence $\rightarrow$ Colloc$^+$
Word $\rightarrow$ SyllableIF
Word $\rightarrow$ SyllableI Syllable SyllableF
OnsetI $\rightarrow$ Consonant$^+$
Nucleus $\rightarrow$ Vowel$^+$

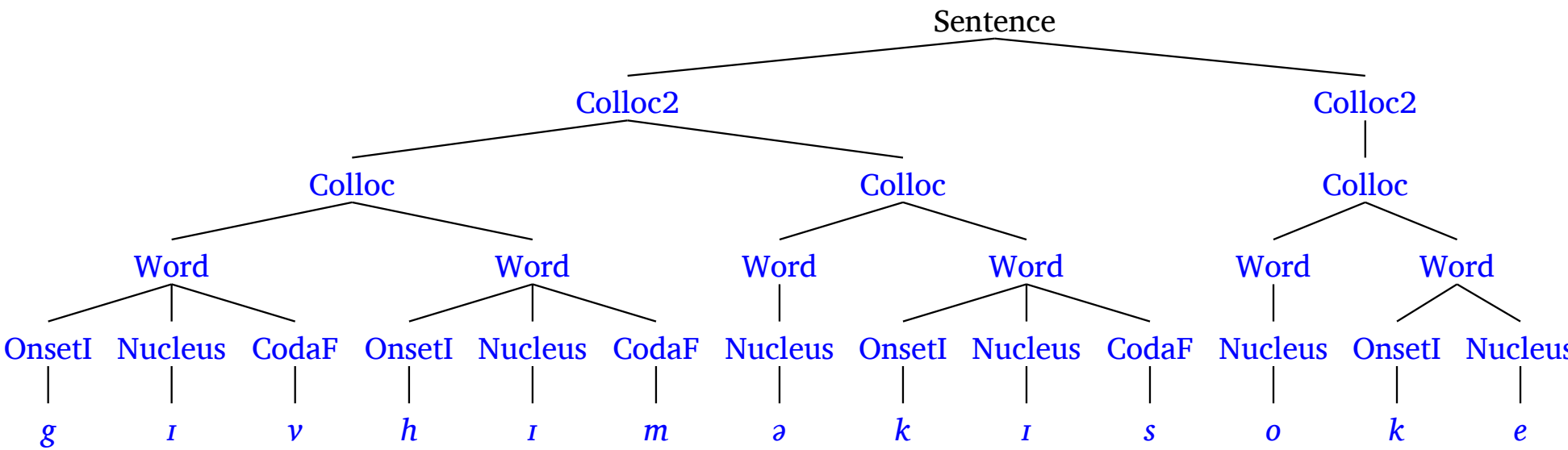
Colloc $\rightarrow$ Word$^+$
Word $\rightarrow$ SyllableI SyllableF
SyllableIF $\rightarrow$ (OnsetI) RhymeF
RhymeF $\rightarrow$ Nucleus (CodaF)
CodaF $\rightarrow$ Consonant$^+$

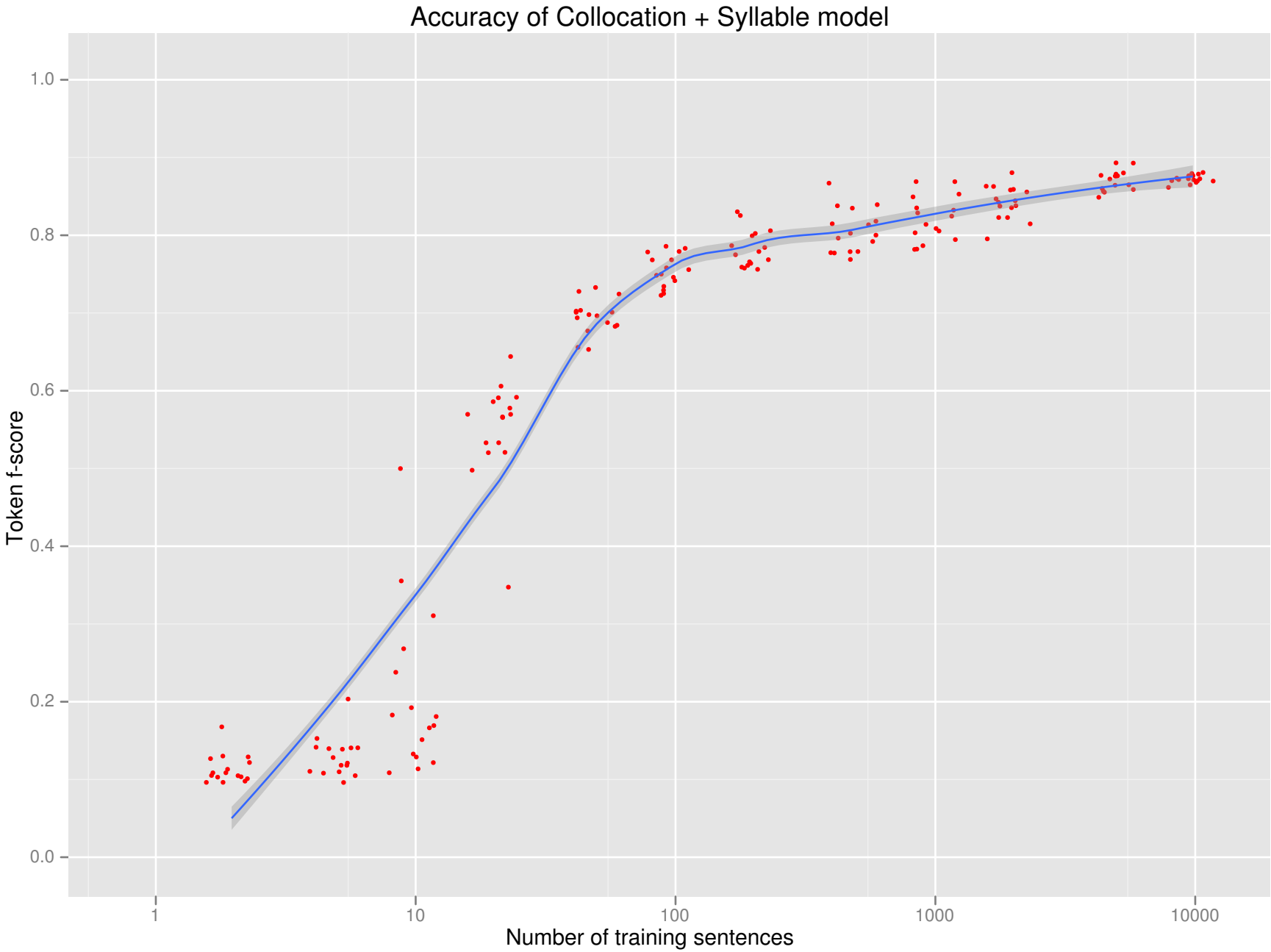


- With 2 Collocation levels, not distinguishing initial/final clusters, f-score = 84%
- With 3 Collocation levels, distinguishing initial/final clusters, f-score = 87%

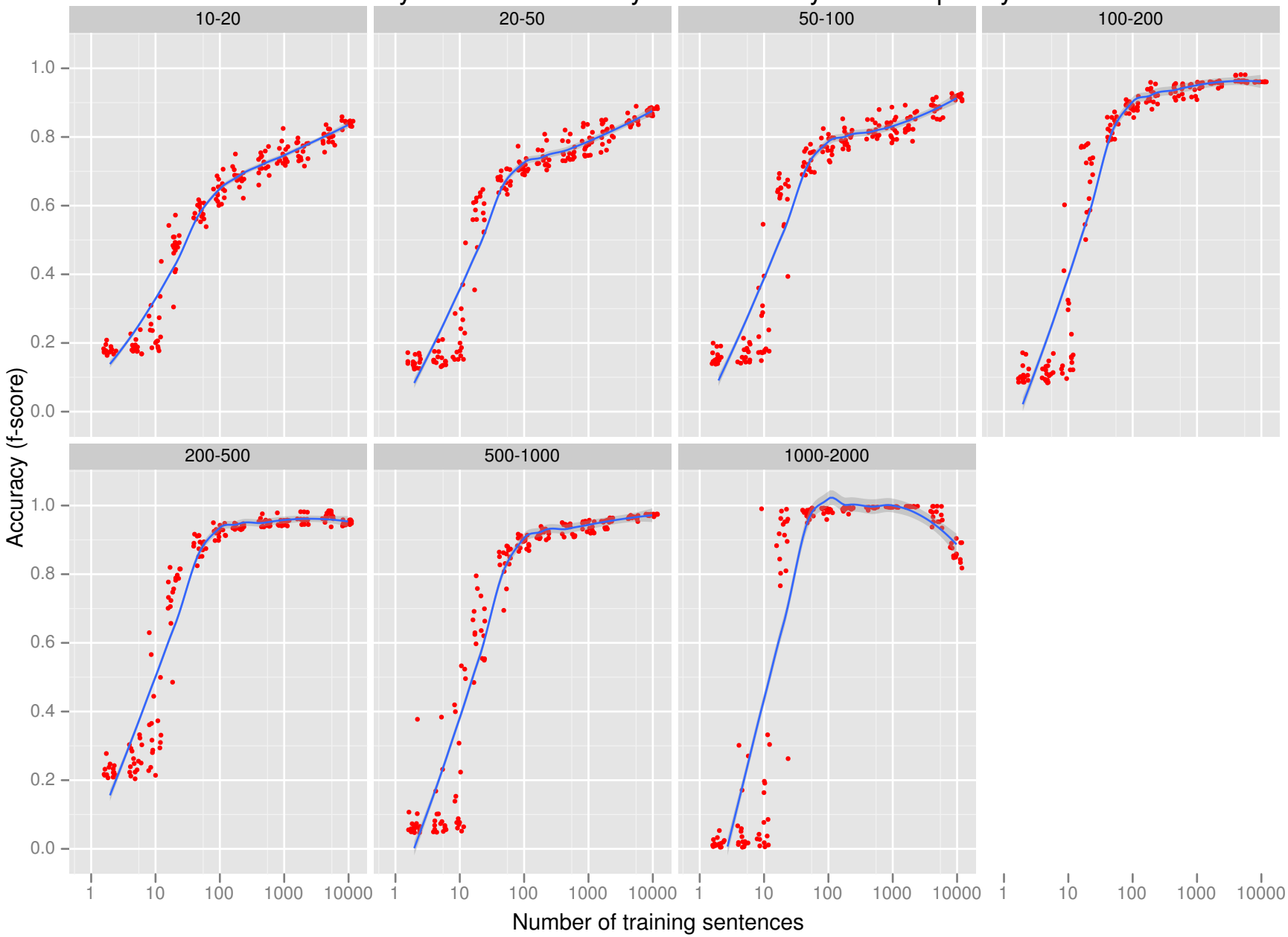
MACQUARIE UNIVERSITY

Collocations^2 $\Rightarrow$ Words $\Rightarrow$ Syllables





MACQUARIE UNIVERSITY



MACQUARIE UNIVERSITY

Summary of English word segmentation

- Word segmentation accuracy depends on the kinds of generalisations learnt.

Generalization	**Accuracy**
words as units (unigram)	56%
+ associations between words (collocations)	76%
+ syllable structure	84%
+ interaction between segmentation and syllable structure	87%

- *Synergies in learning words and syllable structure*
  - joint inference permits the learner to *explain away* potentially misleading generalizations

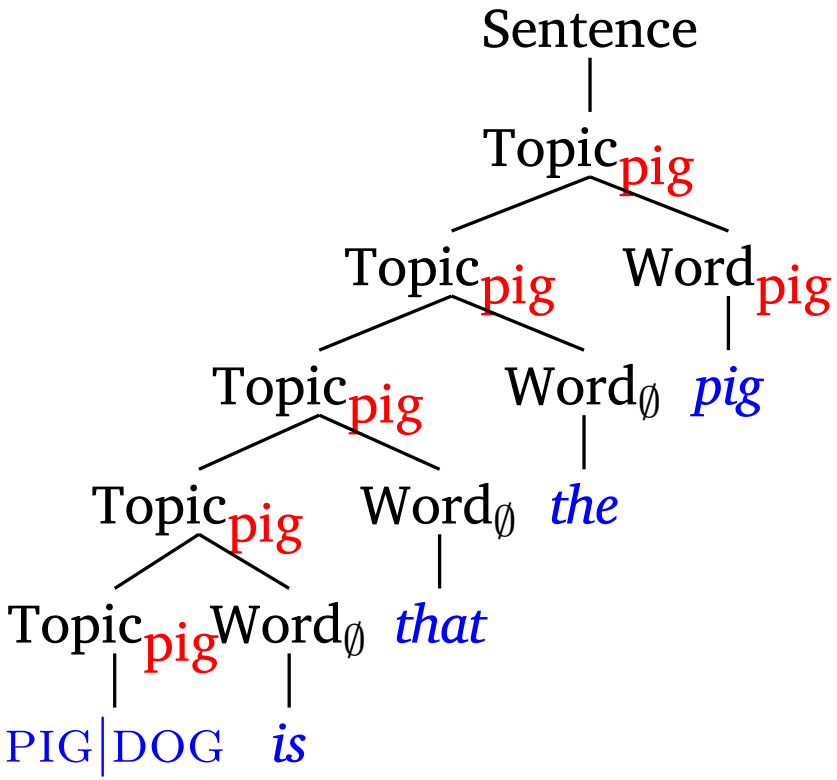
Outline

Prior work: mapping words to topics

- Input to learner:
  - word sequence: *Is that the pig?*
  - objects in nonlinguistic context: DOG, PIG
- Learning objectives:
  - identify utterance topic: PIG
  - identify word-topic mapping: *pig* $\mapsto$ PIG

Frank et al (2009) "topic models" as PCFGs

- Prefix sentences with *possible topic marker*, e.g., PIG|DOG
- PCFG rules *choose a topic* from topic marker and *propagate it through sentence*
- Each word is either generated from sentence topic or null topic ∅

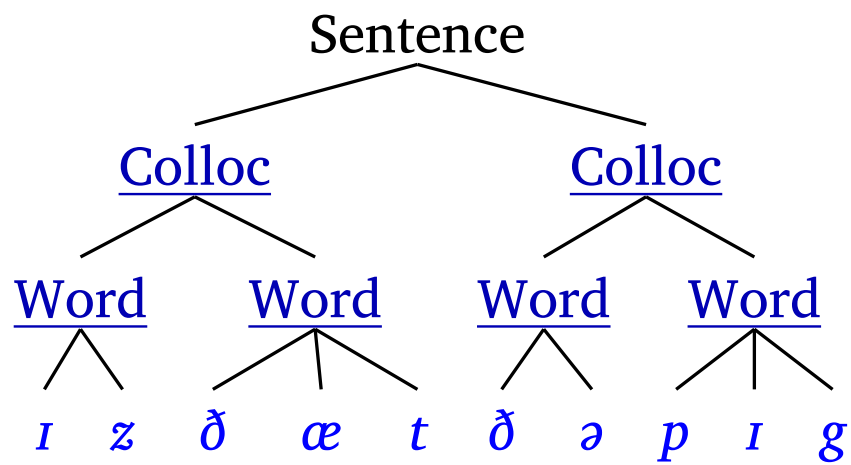


- Grammar can require *at most one topical word per sentence*
- Bayesian inference for PCFG rules and trees corresponds to Bayesian inference for word and sentence topics using topic model (Johnson 2010)

Word segmentation with adaptor grammars

- Adaptor grammars (AGs) can learn the probability of entire subtrees (as well as rules)
- AGs can express several different word segmentation models
- Learning collocations as well as words significantly improves segmentation accuracy

$$\text{Sentence} \rightarrow \underline{\text{Colloc}}^{+}$$
$$\underline{\text{Colloc}} \rightarrow \underline{\text{Word}}^{+}$$
$$\underline{\text{Word}} \rightarrow \text{Phon}^{+}$$

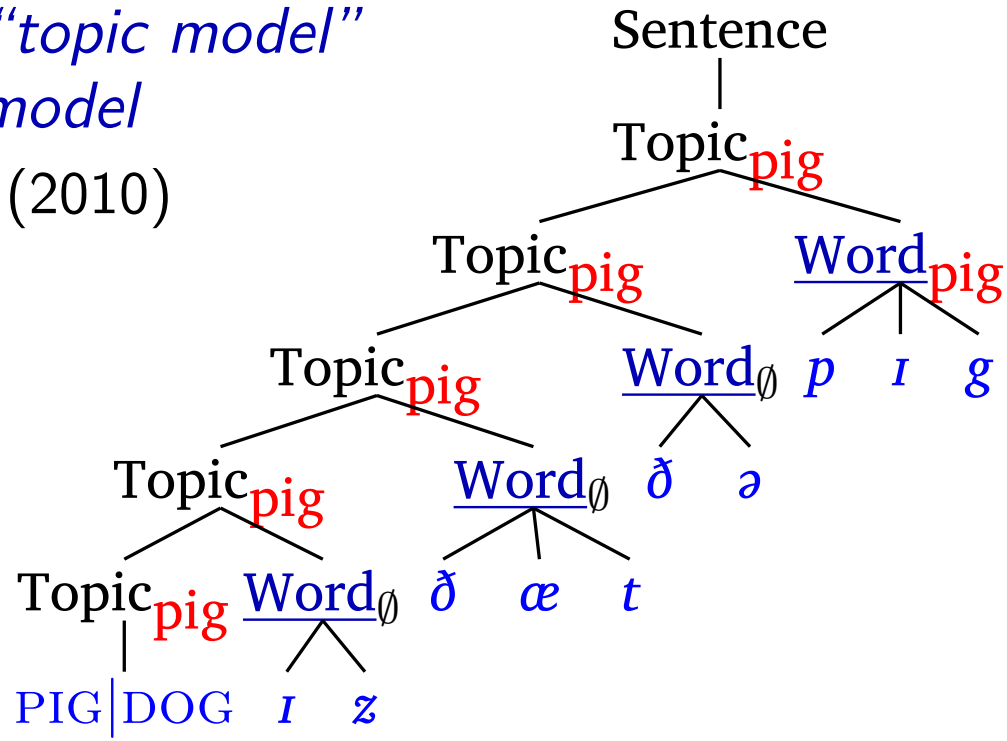


AGs for joint segmentation and topic-mapping

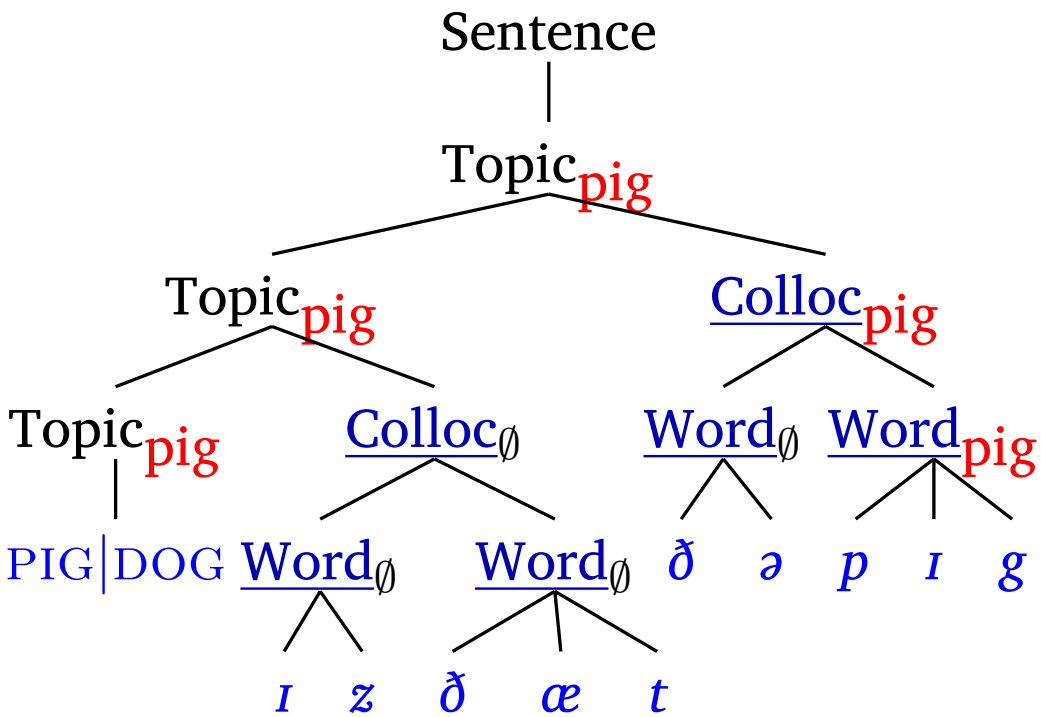
- Combine topic-model PCFG with word segmentation AGs
- Input consists of unsegmented phonemic forms prefixed with possible topics:

PIG|DOG ɪ z ð æ t ð ə p ɪ g

- E.g., combination of *Frank "topic model"* and *unigram segmentation model*
  - equivalent to Jones et al (2010)
- Easy to define *other combinations of topic models and segmentation models*



Collocation topic model AG



- Collocations are either "topical" or not
- Easy to modify this grammar so
  - at most one topical word per sentence, or
  - at most *one topical word per topical collocation*

Experimental set-up

- Input consists of unsegmented phonemic forms prefixed with possible topics:

$$\text{PIG}|\text{DOG}\ \imath\ z\ \eth\ æ\ t\ \eth\ ə\ p\ \imath\ g$$

  - Child-directed speech corpus collected by Fernald et al (1993)
  - Objects in visual context annotated by Frank et al (2009)
- Bayesian inference for AGs using MCMC (Johnson et al 2009)
  - Uniform prior on PYP *a* parameter
  - "Sparse" Gamma$(100, 0.01)$ on PYP *b* parameter
- For each grammar we ran 8 MCMC chains for 5,000 iterations
  - collected word segmentation and topic assignments at every 10th iteration during last 2,500 iterations
    $\Rightarrow$ 2,000 sample analyses per sentence
  - computed and evaluated the modal (i.e., most frequent) sample analysis of each sentence

Does non-linguistic context help segmentation?

Model		word segmentation
segmentation	**topics**	**token f-score**
unigram	not used	0.533
unigram	any number	0.537
unigram	one per sentence	0.547
collocation	not used	0.695
collocation	any number	0.726
collocation	one per sentence	0.719
collocation	one per collocation	**0.750**

- Not much improvement with unigram model
  - consistent with results from Jones et al (2010)
- Larger improvement with collocation model
  - most gain with *one topical word per topical collocation* (this constraint cannot be imposed on unigram model)

Does better segmentation help topic identification?

- Task: identify object (if any) *this sentence* is about

Model		sentence topic	
segmentation	**topics**	**accuracy**	**f-score**
unigram	not used	0.709	0
unigram	any number	0.702	0.355
unigram	one per sentence	0.503	0.495
collocation	not used	0.709	0
collocation	any number	0.728	0.280
collocation	one per sentence	0.440	0.493
collocation	one per collocation	**0.839**	**0.747**

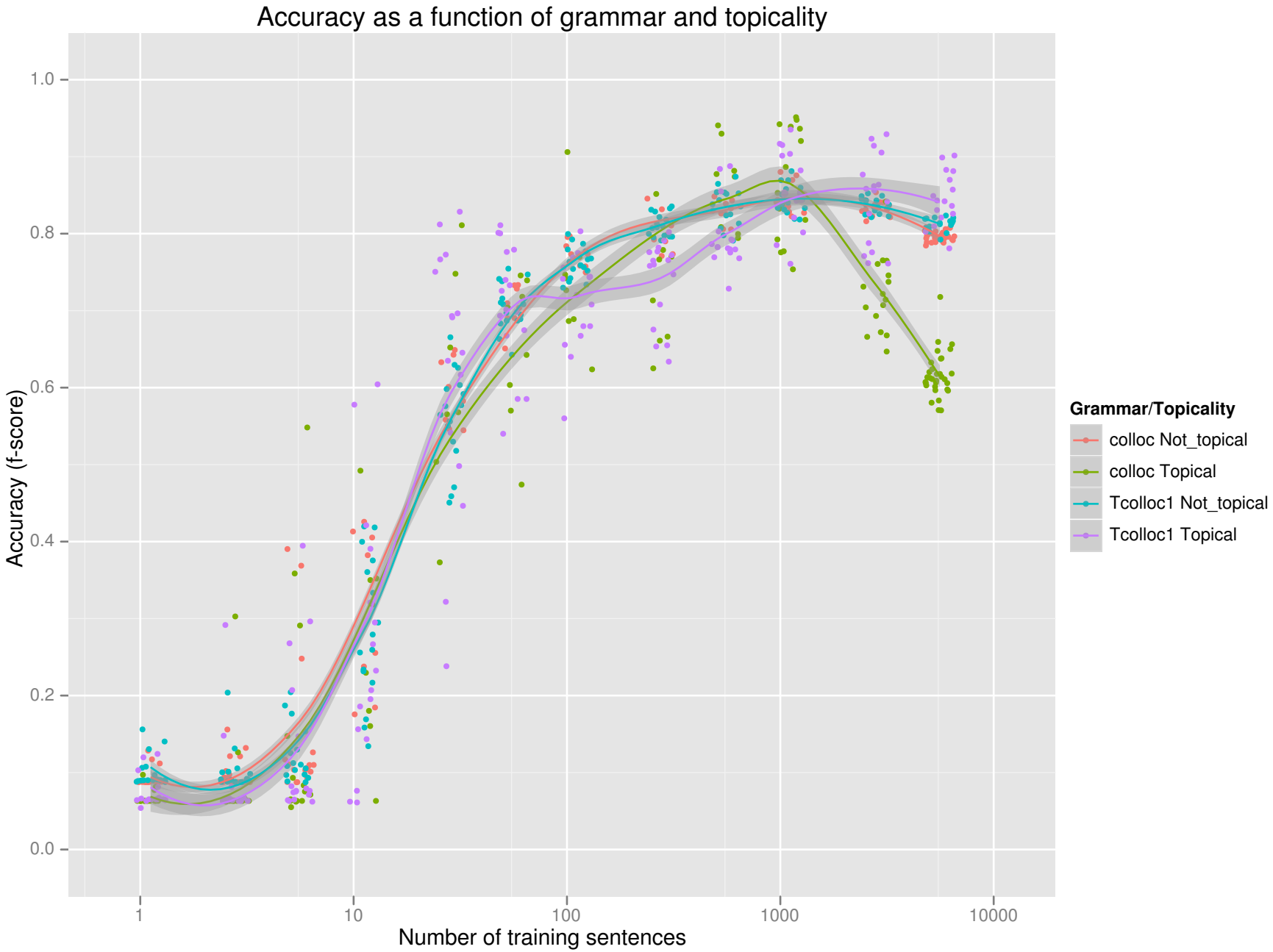
- The collocation grammar with *one topical word per topical collocation* is the only model clearly better than baseline

Does better segmentation help learning word-topic mappings?

- Task: identify *head nouns* of NPs referring to topical objects (e.g. *pɪg* $\mapsto$ PIG in input PIG | DOG ɪ z ð æ t ð ə p ɪ g)

Model		topical word
segmentation	**topics**	**f-score**
unigram	not used	0
unigram	any number	0.149
unigram	one per sentence	0.147
collocation	not used	0
collocation	any number	0.220
collocation	one per sentence	0.321
collocation	one per collocation	**0.636**

- The collocation grammar with one topical word per topical collocation is best at identifying head nouns of topical NPs



MACQUARIE UNIVERSITY

Summary of jointly learning word segmentation and word-to-topic mappings

- *Word to object mapping is learnt more accurately when words are segmented more accurately*
  - improving segmentation accuracy improves topic detection and acquisition of topical words
- *Word segmentation accuracy improves when exploiting non-linguistic context information*
  - incorporating word-topic mapping improves segmentation accuracy (at least with collocation grammars)

$\Rightarrow$ *There are synergies a learner can exploit when learning word segmentation and word-object mappings*

- Limitation: model handles topic dependencies by "feature-passing", but atomic labels have limited ability to handle richly-structured topics

Outline

MACQUARIE UNIVERSITY

Conclusions and future work

- *Joint learning* often uses information in the input more effectively than staged learning
  - Learning syllable structure and word segmentation
  - Learning word-topic associations and word segmentation
- *Do children exploit such synergies in language acquisition?*
- Adaptor grammars are a flexible framework for stating non-parametric hierarchical Bayesian models
  - the accuracies obtained here are the best reported in the literature
- Future work: make the models more realistic
  - extend expressive power of AGs (e.g., feature-passing)
  - richer data (e.g., more non-linguistic context)
  - more realistic data (e.g., stress, phonological variation)

How specific should our computational models be?

- **Marr's (1982) three levels of computational models:**
  - *computational level* (inputs, outputs and relation between them)
  - *algorithmic level* (steps involved in mapping from input to output)
  - *implementation level* (physical processes involved)
- Algorithmic-level models are extremely popular, but I think we should focus on computational-level models first
  - we know almost nothing about *how hierarchical structures are represented and manipulated in the brain*

  $\Rightarrow$ we know almost nothing about which data structures and operations are neurologically plausible
  - current models only explain a tiny fraction of language processing or acquisition
  - typically computational models can be extended, while algorithms need to be completely changed

  $\Rightarrow$ *today's computational models have a greater chance of being relevant than today's algorithms*

Why a child's learning algorithm may be nothing like our algorithms

- Enormous differences in "hardware" $\Rightarrow$ different feasible algorithms
- As scientists we need *generic learning algorithms*, but a child only needs a *specific learning algorithm*
  - as scientists we want to study the effects of different modelling assumptions on learning

  $\Rightarrow$ we need generic algorithms that work for a range of different models, so we can compare them
  - a child only needs an algorithm that works for whatever model they have

  $\Rightarrow$ the child's algorithm might be specialised to their model, and need not work at all for other kinds of models
- The field of *machine learning* has developed many generic learning algorithms: Expectation-maximisation, variational Bayes, Markov chain Monte Carlo, Gibbs samplers, particle filters, ...

The future of Bayesian models of language acquisition

$$\underbrace{\mathrm{P}(\mathrm{Grammar} \mid \mathrm{Data})}_{\text{Posterior}} \propto \underbrace{\mathrm{P}(\mathrm{Data} \mid \mathrm{Grammar})}_{\text{Likelihood}} \underbrace{\mathrm{P}(\mathrm{Grammar})}_{\text{Prior}}$$

- So far our grammars and priors don't encode much linguistic knowledge, but in principle they can!
  - how do we represent this knowledge?
  - how can we learn efficiently using this knowledge?
- Should permit us to *empirically investigate effects of specific universals on the course of language acquisition*
- My guess: the interaction between innate knowledge and learning will be *richer and more interesting* than either the rationalists or empiricists currently imagine!