Adaptor Grammars for Learning Non-Concatenative Morphology

Jan Botha Phil Blunsom

Department of Computer Science
University of Oxford

EMNLP 2013, Seattle

Morphology, the usual:

talk
talks
talked
talking

Morphology, the usual:

talk
talks
talked
talking

Morphology, the usual:

talk
talks
talked
talking

Gesundheitsreform *(health reform)* [German]

Morphology, the usual:

talk
talks
talked
talking

Gesundheitsreform *(health reform)* [German]

Morphology, the usual:

talk
talks
talked
talking

Gesundheitsreform *(health reform)* [German]

→ Permits concatenative view.

Morphology, the usual:

talk
talks
talked
talking

Gesundheitsreform *(health reform)* [German]

→ Permits concatenative view.

The “other” morphology:

kitAb	*(book)*
kutub	*(books)*

[Arabic]

Morphology, the usual:

talk
talks
talked
talking

Gesundheitsreform *(health reform)* [German]

→ Permits concatenative view.

The “other” morphology:

kitAb *(book)*
kutub *(books)* [Arabic]

Morphology, the usual:

talk
talks
talked
talking

Gesundheitsreform *(health reform)* [German]

→ Permits concatenative view.

The “other” morphology:

kitAb	*(book)*	[Arabic]
kutub	*(books)*	
wakitAbi	*(and my book)*	

Morphology, the usual:

talk
talks
talked
talking

Gesundheitsreform *(health reform)* [German]

→ Permits concatenative view.

The “other” morphology:

kitAb	*(book)*
kutub	*(books)*
wakitAbi	*(and my book)*

[Arabic]

→ Needs non-concatenative view.

Morphology, the usual:

talk
talks
talked
talking

Gesundheitsreform *(health reform)* [German]

→ Permits concatenative view.

The “other” morphology:

kitAb	*(book)*
kutub	*(books)*
wakitAbi	*(and my book)*

[Arabic]

→ Needs non-concatenative view.

⇒ This talk deals with **both** views; focus on Arabic & Hebrew.

Motivation

Rich morphology $\Rightarrow$ novelty, sparse data

Motivation

Rich morphology $\Rightarrow$ novelty, sparse data

Rule-based & supervised methods

- the usual limitations – costly, language-dependent ...

Motivation

Rich morphology $\Rightarrow$ novelty, sparse data

Rule-based & supervised methods

- the usual limitations – costly, language-dependent ...

Unsupervised methods

- mostly limited to concatenative morphology, or:
- constrained search + dictionaries
  (Darwish, 2002; Boudlal et al., 2009)
- statistics + heuristic constraints
  (Rodrigues & Cavar, 2007; Daya et al., 2008)
- nonparametric Bayesian model (Fullwood & O'Donnel, 2013)

Motivation

Rich morphology $\Rightarrow$ novelty, sparse data

Rule-based & supervised methods

- the usual limitations – costly, language-dependent ...

Unsupervised methods

- mostly limited to concatenative morphology, or:
- constrained search + dictionaries
  (Darwish, 2002; Boudlal et al., 2009)
- statistics + heuristic constraints
  (Rodrigues & Cavar, 2007; Daya et al., 2008)
- nonparametric Bayesian model (Fullwood & O'Donnel, 2013)

Aim:

flexible model of joint segmentation and stem formation

The Plan

Proposal:

- encode (non-)concatenative morphology
  with a nice grammar formalism
- cast it as flexible nonparametric Bayesian model

The Plan

Proposal:

- encode (non-)concatenative morphology with a nice grammar formalism
- cast it as flexible nonparametric Bayesian model

Outcomes:

- improved segmentation of Hebrew & Arabic
- induced lexicons of roots

The Plan

Proposal:

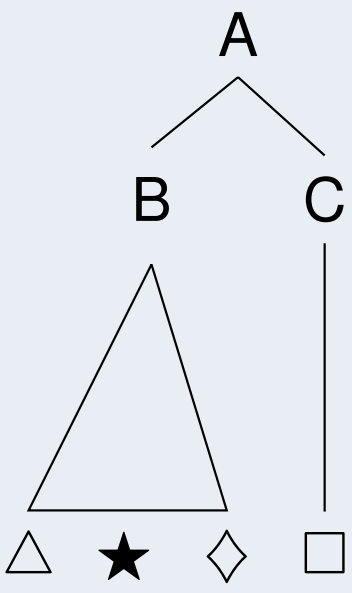
- encode (non-)concatenative morphology with a nice grammar formalism
- cast it as flexible nonparametric Bayesian model

Outcomes:

- improved segmentation of Hebrew & Arabic
- induced lexicons of roots

Context-free Grammars

CFG

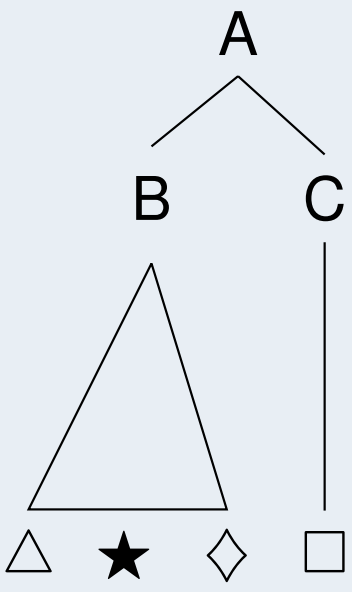


$A \rightarrow B\ C$

Simple Range Concatenating Grammars

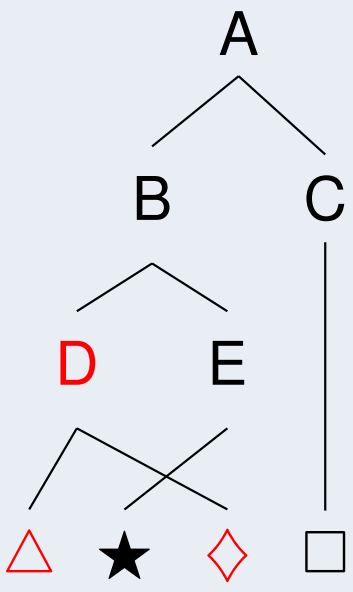
(Boullier, 2000)

CFG



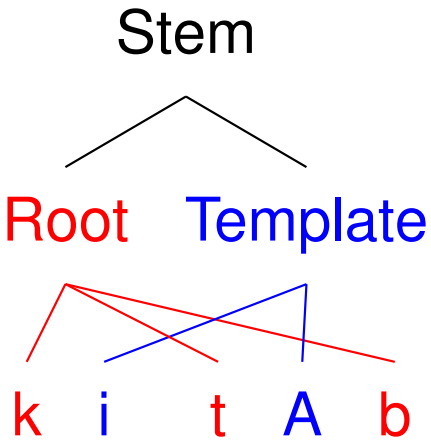
$A \rightarrow B\ C$

SRCG



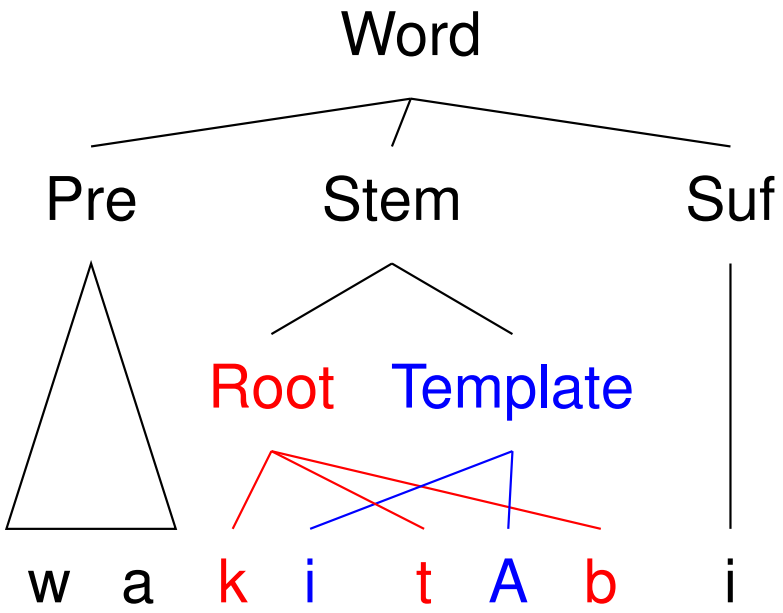
$B(xyz) \rightarrow D(x,z)\ E(y)$

MODELLING MORPHOLOGY WITH SRCGS



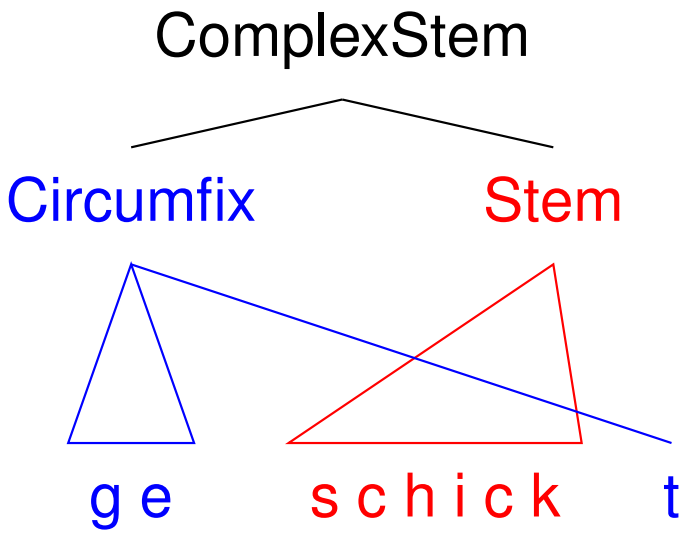
"book"

Modelling Morphology with SRCGs



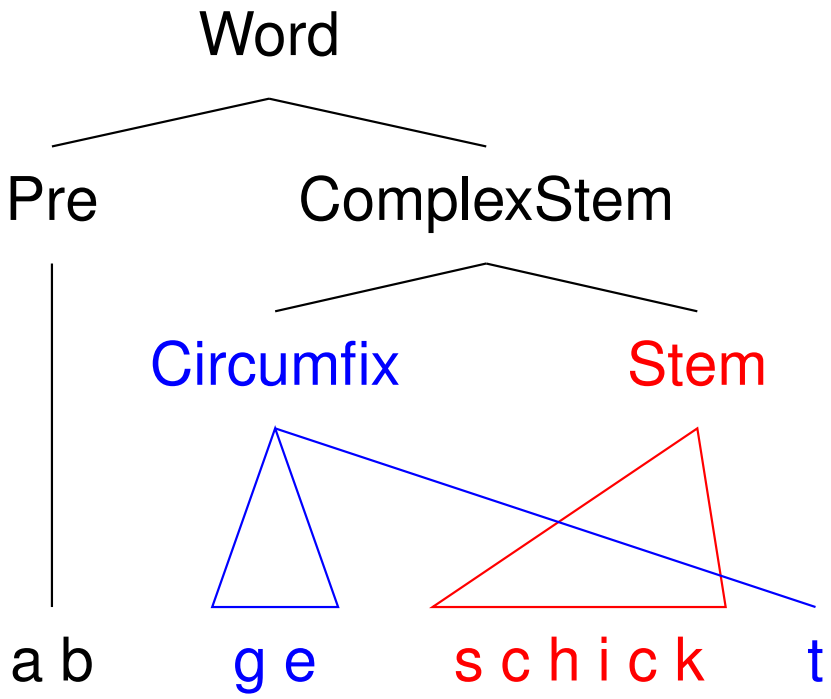
“and my book”

MODELLING MORPHOLOGY WITH SRCGS



“sent”

Modelling Morphology with SRCGs



“sent off”

Probabilistic Grammar

SRCGs provide the sought-after linguistic flexibility.

Probabilistic Grammar

SRCGs provide the sought-after linguistic flexibility.

Add **Adaptor Grammars** (Johnson et al., 2007)

- nonparametric Bayesian extension of PCFGs

Probabilistic Grammar

SRCGs provide the sought-after linguistic flexibility.

Add **Adaptor Grammars** (Johnson et al., 2007)

- nonparametric Bayesian extension of PCFGs
- whole subtrees reused across different inputs
  e.g. stem occurring in multiple word forms

Probabilistic Grammar

SRCGs provide the sought-after linguistic flexibility.

Add **Adaptor Grammars** (Johnson et al., 2007)

- nonparametric Bayesian extension of PCFGs
- whole subtrees reused across different inputs
  e.g. stem occurring in multiple word forms
- Expanding a node (generatively) involves:
  a) generate its direct children via base grammar
  OR
  b) pick a completed subtree from cache.

Probabilistic Grammar

SRCGs provide the sought-after linguistic flexibility.

Add **Adaptor Grammars** (Johnson et al., 2007)

- nonparametric Bayesian extension of PCFGs
- whole subtrees reused across different inputs
  e.g. stem occurring in multiple word forms
- Expanding a node (generatively) involves:
  a) generate its direct children via base grammar
  OR
  b) pick a completed subtree from cache.
- choice governed by Pitman-Yor Process

Probabilistic Grammar

SRCGs provide the sought-after linguistic flexibility.

Add **Adaptor Grammars** (Johnson et al., 2007)

- nonparametric Bayesian extension of PCFGs
- whole subtrees reused across different inputs
  e.g. stem occurring in multiple word forms
- Expanding a node (generatively) involves:
  a) generate its direct children via base grammar
  OR
  b) pick a completed subtree from cache.
- choice governed by Pitman-Yor Process

We apply this to SRCGs.

Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

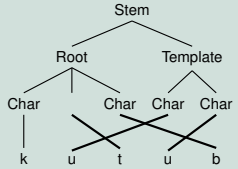
$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

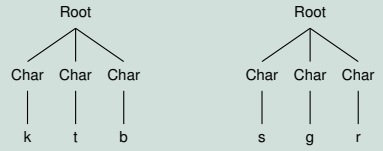
$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

Cache

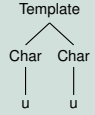
Stem



Root



Template



Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

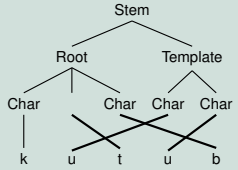
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

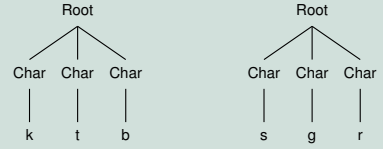
Word

Cache

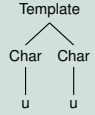
Stem



Root



Template



Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

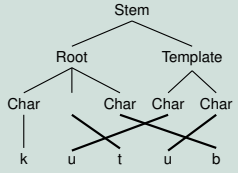
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

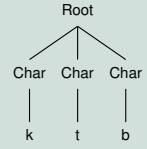
Word

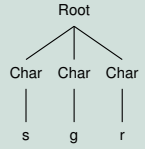
Cache

Stem

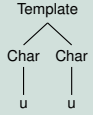


Root





Template



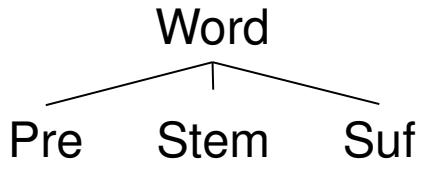
Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

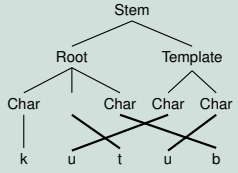
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

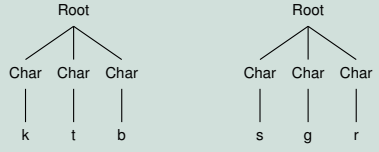


Cache

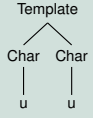
Stem



Root



Template



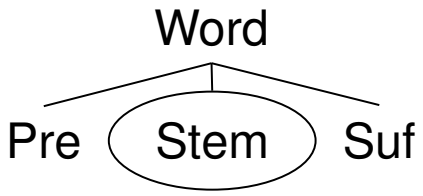
Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

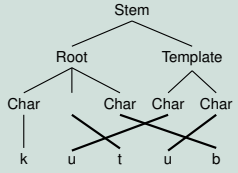
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

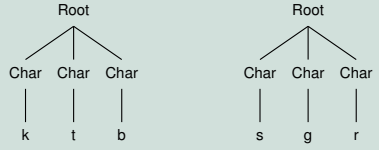


Cache

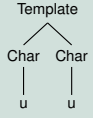
Stem



Root



Template



Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

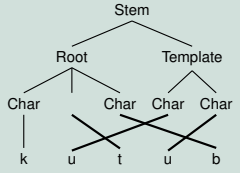
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

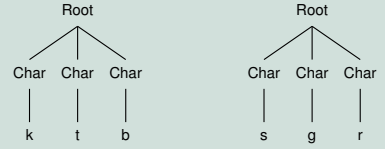
Stem

Cache

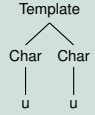
Stem



Root



Template



Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

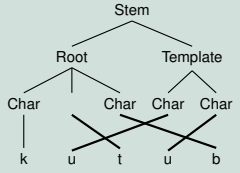
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

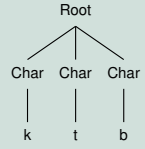
Stem

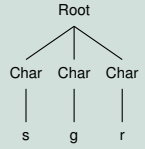
Cache

Stem

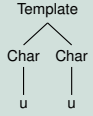


Root





Template



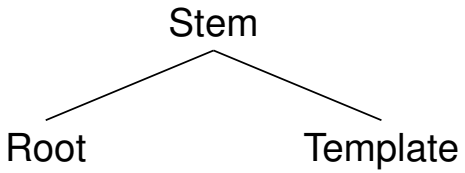
Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

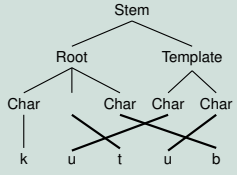
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

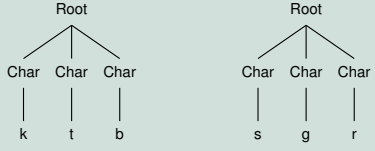


Cache

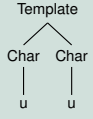
Stem



Root



Template



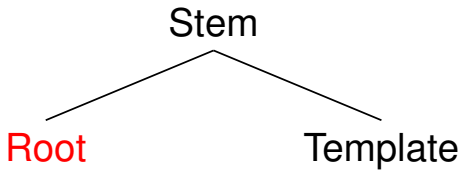
Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

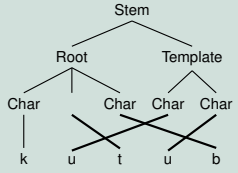
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

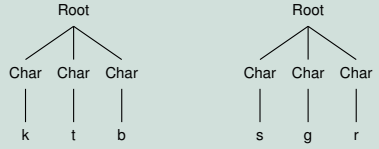


Cache

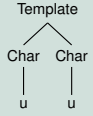
Stem



Root



Template



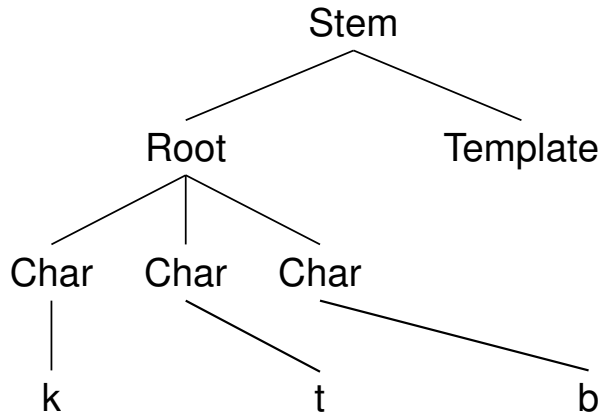
Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

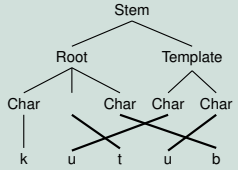
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

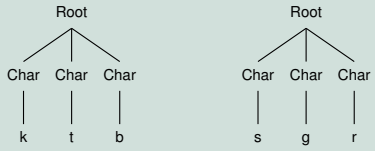


Cache

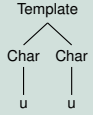
Stem



Root



Template



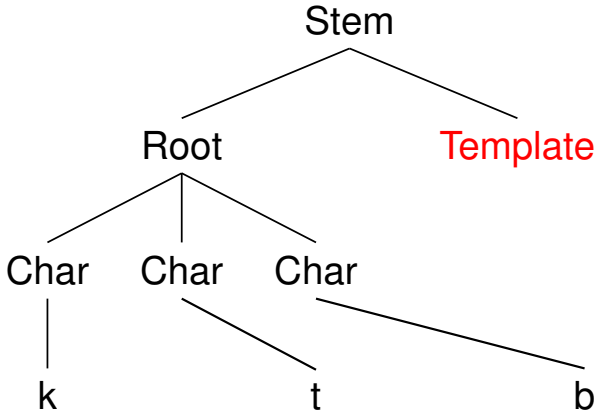
Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

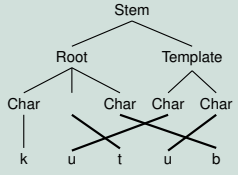
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

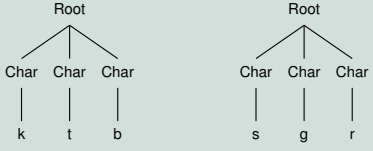


Cache

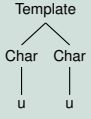
Stem



Root



Template



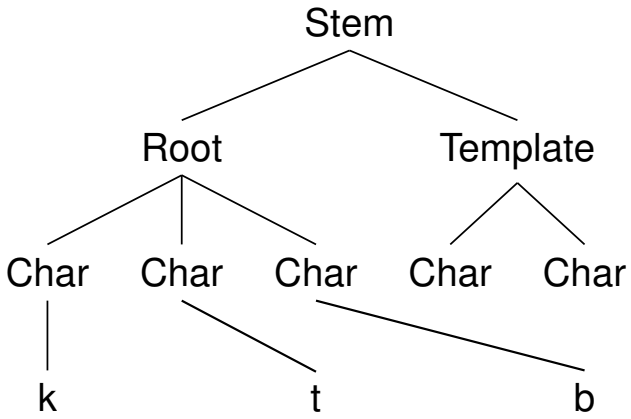
Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

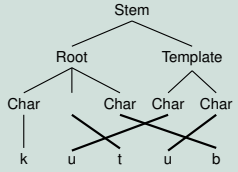
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

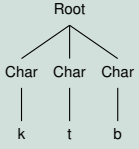


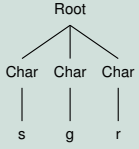
Cache

Stem

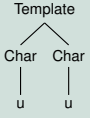


Root





Template



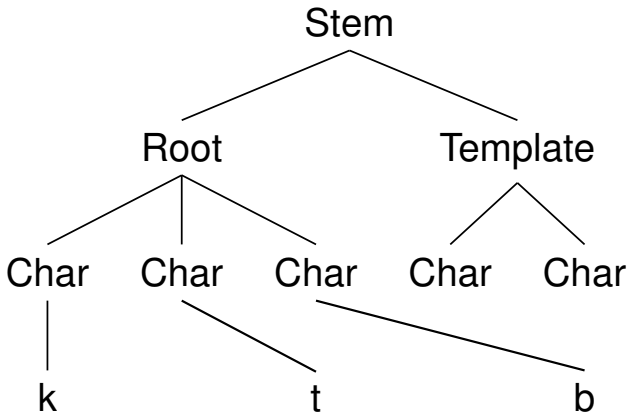
Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

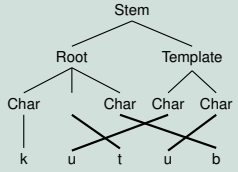
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

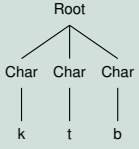


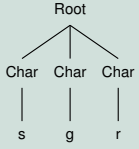
Cache

Stem

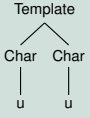


Root





Template



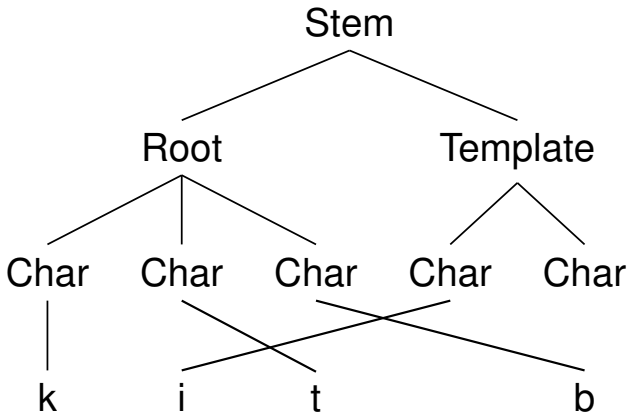
Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

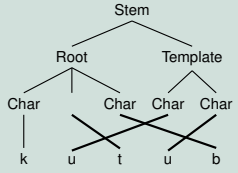
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

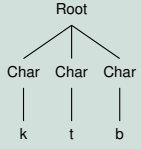


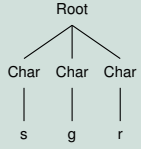
Cache

Stem

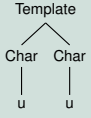


Root





Template



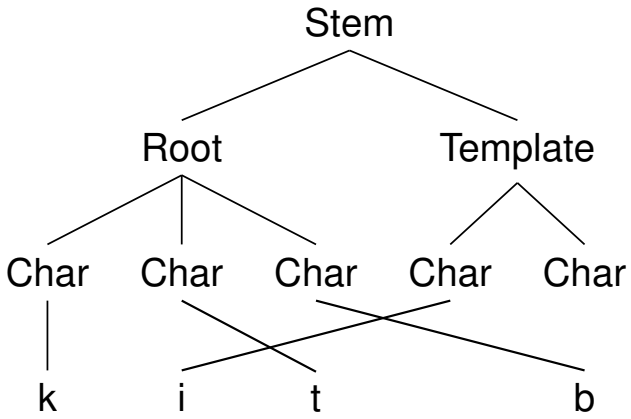
Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

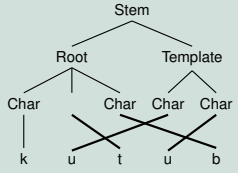
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

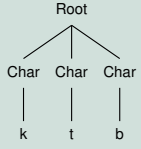


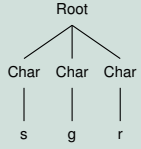
Cache

Stem

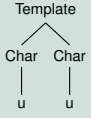


Root





Template



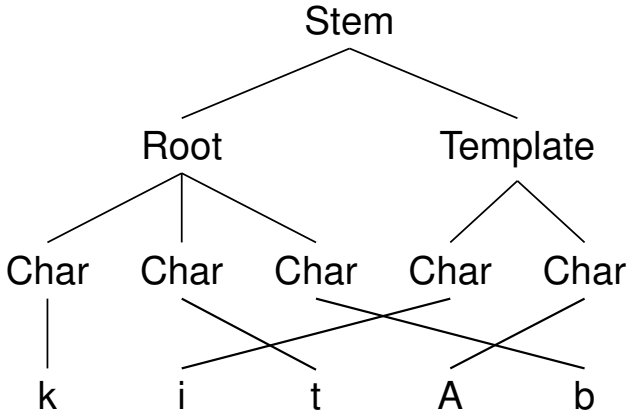
Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

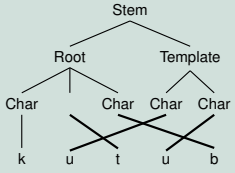
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$

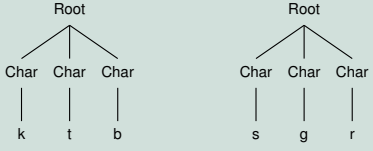


Cache

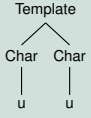
Stem



Root



Template



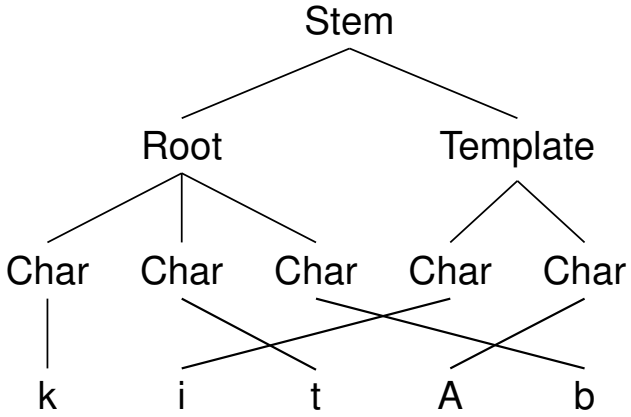
Grammar Fragment

$$\text{Word}(xyz) \rightarrow \underline{\text{Pre}}(x)\ \underline{\text{Stem}}(y)\ \underline{\text{Suf}}(z)$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{Root}}(a,c,e)\ \underline{\text{Template}}(b,d)$$

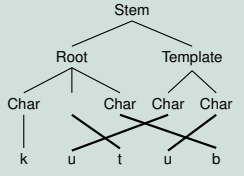
$$\underline{\text{Root}}(f,g,h) \rightarrow \text{Char}(f)\ \text{Char}(g)\ \text{Char}(h)$$

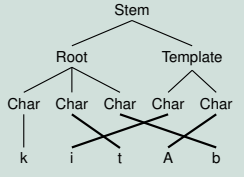
$$\underline{\text{Template}}(i,j) \rightarrow \text{Char}(i)\ \text{Char}(j)$$



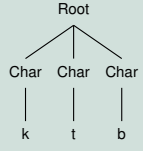
Cache

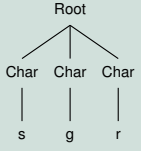
Stem



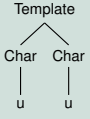


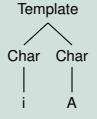
Root





Template





DATASETS

Hebrew

- 5k word types from CHILDES database
- vocalised

Standard Arabic $\times 2$

- synthesised 50k word types from BAMA dictionaries
- both orthographic variants: vocalised and unvocalised

Quranic Arabic

- 18k word types from annotated Quran (Dukes et al. 2010)
- extensive diacritics

Included all parts of speech; not filtered for verbs/nouns.

Method

1. Run MCMC sampler over unannotated data.
2. Collect 100 posterior samples.
3. Evaluate MAP parses against references.

Method

1. Run MCMC sampler over unannotated data.
2. Collect 100 posterior samples.
3. Evaluate MAP parses against references.

Task 1: Segmentation

$$\text{ions} \rightarrow \text{ion} \cdot \text{s}$$

$$\text{fabrications} \rightarrow \text{fabricat} \cdot \text{ion} \cdot \text{s}$$

Method

1. Run MCMC sampler over unannotated data.
2. Collect 100 posterior samples.
3. Evaluate MAP parses against references.

Task 1: Segmentation

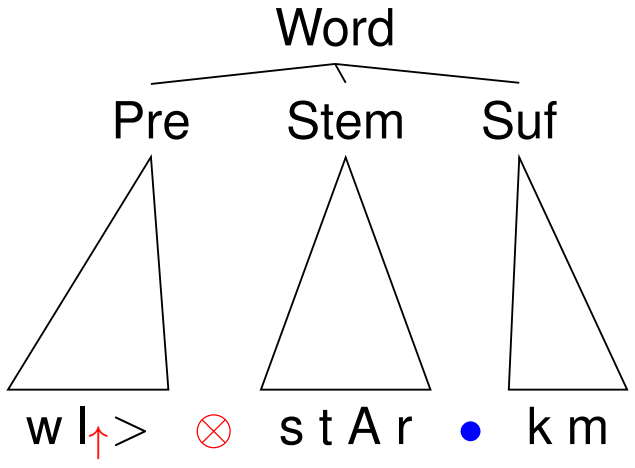
$$\text{ions} \rightarrow \text{ion} \cdot \text{s}$$
$$\text{fabrications} \rightarrow \text{fabricat} \cdot \text{ion} \cdot \text{s}$$

Task 2: Lexicon Induction

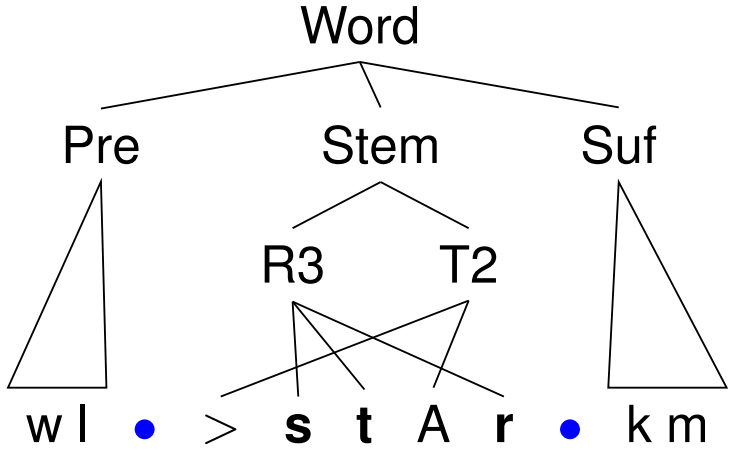
Prefixes	**Stems**	**Suffixes**	**Roots**
...	ion	ion	...
	fabricat	s	

EXAMPLE ANALYSES – ARABIC (UNVOCALISED)

CFG

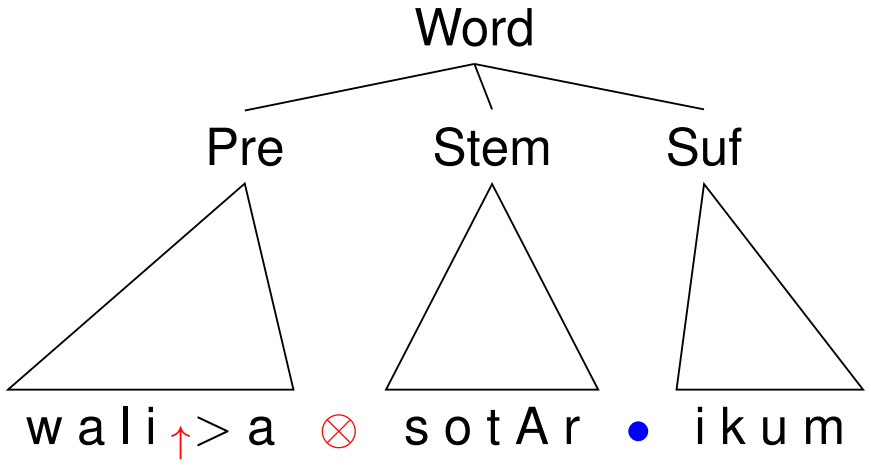


SRCG

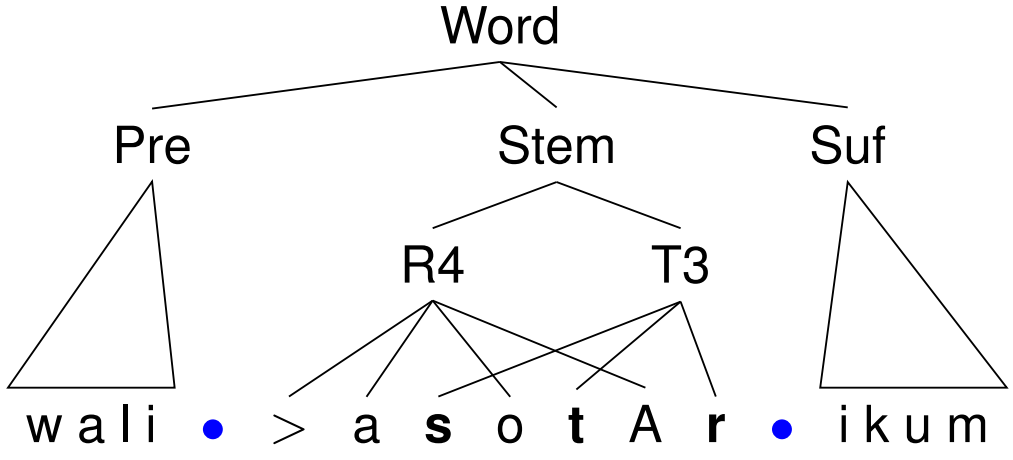


Example Analyses – Arabic (vocalised)

CFG

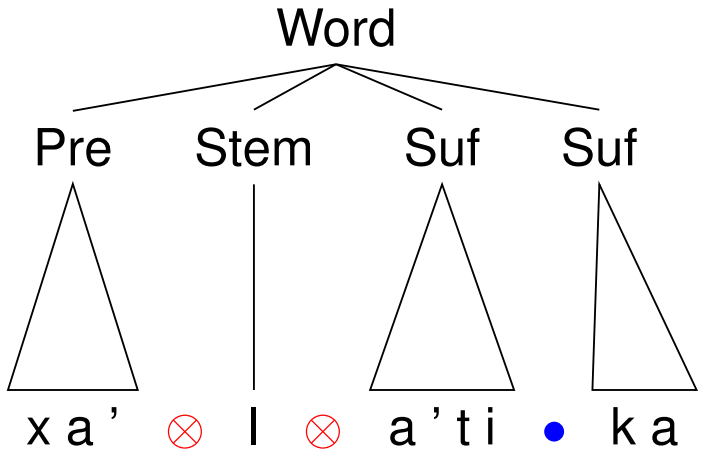


SRCG

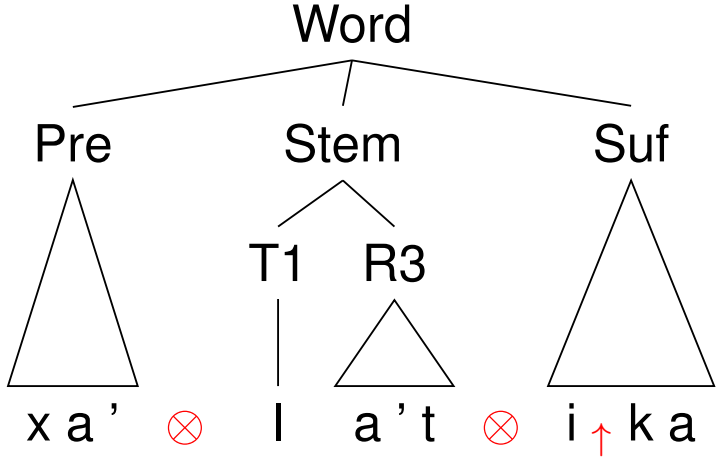


Example Analyses – Quran

CFG



SRCG



Example Analyses

Top three Hebrew roots according to model

	Root
1.	**spr** *(tell)*
2.	**lbš** *(wear)*
3.	**ptx** *(open)*

Example Analyses

Top three Hebrew roots according to model

	Root	Correct Instances	
1.	**spr** *(tell)*	**s**i**p**a**r**•ti	ye•**s**a**pr**•u
2.	**lbš** *(wear)*	li•**lb**o**š**	ti•**lb**e**š**•i
3.	**ptx** *(open)*		

Example Analyses

Top three Hebrew roots according to model

	Root	Correct Instances		Mistaken Instance
1.	**spr** *(tell)*	**s**i**p**ar•ti	ye•**s**a**pr**•u	hi⊗**s**ta**p**⊗ar$_{\uparrow}$t
2.	**lbš** *(wear)*	li•**lb**oš	ti•**lb**eš•i	le$_{\uparrow}$ha⊗**lbi**š
3.	**ptx** *(open)*			

Segmentation Results

F1-scores over word-internal morpheme boundaries

(higher better)

Morfessor
CFG
SRCG1
SRCG2
SRCG3
SRCG4

Segmentation Results

F1-scores over word-internal morpheme boundaries

(higher better)

	Arabic (unvoc)
Morfessor	55.6
CFG	47.4
SRCG1	60.4
SRCG2	60.5
SRCG3	64.5
SRCG4	**74.5**

Segmentation Results

F1-scores over word-internal morpheme boundaries

(higher better)

	Arabic (unvoc)	**Arabic**
Morfessor	55.6	40.0
CFG	47.4	64.2
SRCG1	60.4	71.9
SRCG2	60.5	72.2
SRCG3	64.5	71.6
SRCG4	**74.5**	**73.7**

Segmentation Results

F1-scores over word-internal morpheme boundaries

(higher better)

	Arabic (unvoc)	**Arabic**	**Hebrew**
Morfessor	55.6	40.0	24.2
CFG	47.4	64.2	60.1
SRCG1	60.4	71.9	77.3
SRCG2	60.5	72.2	77.4
SRCG3	64.5	71.6	77.1
SRCG4	**74.5**	**73.7**	**78.1**

Segmentation Results

F1-scores over word-internal morpheme boundaries

(higher better)

	Arabic (unvoc)	**Arabic**	**Hebrew**	**Quran**
Morfessor	55.6	40.0	24.2	**44.3**
CFG	47.4	64.2	60.1	19.6
SRCG1	60.4	71.9	77.3	22.5
SRCG2	60.5	72.2	77.4	25.7
SRCG3	64.5	71.6	77.1	24.8
SRCG4	**74.5**	**73.7**	**78.1**	-

Segmentation Results

F1-scores over word-internal morpheme boundaries

(higher better)

	Arabic (unvoc)	**Arabic**	**Hebrew**	**Quran**
Morfessor	55.6	40.0	24.2	**44.3**
CFG	47.4	64.2	60.1	19.6
SRCG1	60.4	71.9	77.3	22.5
SRCG2	60.5	72.2	77.4	25.7
SRCG3	64.5	71.6	77.1	24.8
SRCG4	**74.5**	**73.7**	**78.1**	-

$\Rightarrow$ Modelling discontiguous substructure improves segmentation

Morpheme Lexicon Induction Results

F-score in set-based evaluation against gold lexicons

(higher better)

	Prefixes	**Stems**	**Suffixes**	**Triliteral Roots**
Arabic (unvocalised)				P / R / F
CFG	33	44	40	–
Best SRCG	53	58	52	51 / 80 / 62

Conclusions

- Flexible modelling framework that handles concatenative *and* non-concatenative morphology
- Accounting for root-templatic stem formation improved segmentation in Hebrew and Arabic
- SRCG-variant of Adaptor Grammars potentially applicable to other SRCGs (parsing, translation)
- Further avenues for inquiry:
  - Induced roots as features in downstream tasks?
  - Success at handling other non-concatenative phenomena?
  - What does discontiguous model do on English, etc.
  - RCG with deletion $\underset{?}{\rightarrow}$ irregular patterns, weak roots?

Thank you.