Distributional Learning of Context-Free Grammars and Simple Context-Free Tree Grammars

吉仲　亮

京都大学情報学研究科

December 19, 2012
最先端構文解析とその周辺

Outline

Grammatical Inference

- Algorithmic learning of formal languages
  - String languages
  - Tree languages ...
- More theoretical rather than heuristic
- Motivations/Applications:
  - Mathematical model of natural language acquisition
  - Grammar extraction from tagged/untagged corpora
  - Biological sequences

Regular Language Learning and CFL Learning

- Fruitful positive results on the learning of Regular languages
  - Query learning
  - PAC learning under the simple distribution
  - Identification in the limit from positive and negative data
  - Learning interesting subclasses from positive data only
- Nice properties of Regular languages
  - Myhill-Nerode Theorem – Canonical DFA

Regular Language Learning and CFL Learning

- Fruitful positive results on the learning of Regular languages
  - Query learning
  - PAC learning under the simple distribution
  - Identification in the limit from positive and negative data
  - Learning interesting subclasses from positive data only
- Nice properties of Regular languages
  - Myhill-Nerode Theorem – Canonical DFA
- Few positive results on CFL learning
  - No nice mathematical properties
  - No canonical automata/grammars

Regular Language Learning and CFL Learning

- Fruitful positive results on the learning of Regular languages
  - Query learning
  - PAC learning under the simple distribution
  - Identification in the limit from positive and negative data
  - Learning interesting subclasses from positive data only
- Nice properties of Regular languages
  - Myhill-Nerode Theorem – Canonical DFA
- Few positive results on CFL learning before this century
  - No nice mathematical properties
  - No canonical automata/grammars
- *Distributional Learning* for CFLs

Distributional Learning

- Substitutable CFLs are identifiable in the limit from positive data only (Clark and Eyraud 2007)
- Query learning of c-deterministic/congruential CFGs (Shirakawa & Yokomori 1995 / Clark 2010)
- Quite rich subclasses are identifiable in the limit from positive data and MQs (Clark et al. 2009, Clark 2010, Yoshinaka 2010, 2011, 2012 etc.)
- PAC learnability of Unambiguous NTS languages (Clark 2006)

Distributional Learning

- Substitutable CFLs are identifiable in the limit from positive data only (Clark and Eyraud 2007)
- Query learning of c-deterministic/congruential CFGs (Shirakawa & Yokomori 1995 / Clark 2010)
- Quite rich subclasses are identifiable in the limit from positive data and MQs (Clark et al. 2009, Clark 2010, Yoshinaka 2010, 2011, 2012 etc.)
- PAC learnability of Unambiguous NTS languages (Clark 2006)

Heuristics has preceded Theory
(Brill et al. 1990, Adriaans 1999, van Zaanen 2000, Klein & Manning 2002, etc.)

Outline

Notation

A Context-Free Grammar

A tuple $\langle \Sigma, V, I, R \rangle$

- Σ: finite set of terminal symbols
- V: finite set of nonterminal symbols
- $I \subseteq V$: set of initial symbols
- R: set of productions

Bottom-up derivation:

- $\alpha N \gamma \Rightarrow \alpha \beta \gamma$ if $N \rightarrow \beta \in P$
- $\mathcal{L}(G, N) = \{ w \in \Sigma^* \mid N \overset{*}{\Rightarrow} w \}$
- $\mathcal{L}(G) = \bigcup_{S \in I} \mathcal{L}(G, S)$

Context

A context is just a pair of strings $l\Box r$ with $l, r \in \Sigma^*$.

- $(l\Box r) \odot u = lur$,
  - $aa\Box bbb \odot aab = aaaabbbb$,

Context

A context is just a pair of strings $l\Box r$ with $l, r \in \Sigma^*$.

- $(l\Box r) \odot u = lur$,
  - $aa\Box bbb \odot aab = aaaabbbb$,
- $L \oslash u = \{\, l\Box r \mid lur \in L \,\}$ and $L \oslash (l\Box r) = \{\, u \mid lur \in L \,\}$,
- Special context $\Box$:
  - $\Box \odot u = u$,
  - $u \in L \iff \Box \in L \oslash u$.

Context

A context is just a pair of strings $l\Box r$ with $l, r \in \Sigma^*$.

- $(l\Box r) \odot u = lur$,
  - $aa\Box bbb \odot aab = aaaabbbb$,
- $L \oslash u = \{\, l\Box r \mid lur \in L \,\}$ and $L \oslash (l\Box r) = \{\, u \mid lur \in L \,\}$,
- Special context $\Box$:
  - $\Box \odot u = u$,
  - $u \in L \iff \Box \in L \oslash u$.

Syntactic congruence

$$u \equiv_L v \quad \text{iff} \quad L \oslash u = L \oslash v$$

Let $[u] = \{\, v \in \Sigma^* \mid v \equiv_L u \,\}$.

Example

For $L = \{ a^n b^n \mid n \geq 0 \}$,

	$\Box$	$a\Box$	$\Box b$	$a\Box bb$	$\Box abb$
λ	1	0	0	0	0
a	0	0	1	1	1
b	0	1	0	0	0
ab	1	0	0	0	0
aab	0	0	1	1	0
$aaabb$	0	0	1	1	0

$$L \oslash aab = L \oslash aaabb = \{\Box b, a\Box bb, \dots, a^k\Box b^{k+1}, \dots\}$$

$$[aab] = \{ a^{k+1} b^k \mid k \geq 1 \}.$$

$$\forall l\Box r,\ l\Box r \odot aaabb \in L \text{ iff } l\Box r \odot aab \in L \text{ iff } l\Box r \odot a^{k+1}b^k \in L.$$

Keywords of Distributional Learning

- Observe, model, exploit the relation between "substrings" and "contexts" (Observation table indexed by strings and contexts)
- Objectivity
- Symmetric approaches

Outline

Congruential CFGs [Clark 2010]

Congruential context-free grammars

For every nonterminal N of G, if $u, v \in \mathcal{L}(G, N)$, then $u \equiv_{\mathcal{L}(G)} v$.

- If G is congruential, and we binarize G, then the result is congruential.
- So we assume productions are all like $N \to PQ$ or $N \to a$.

Examples

- $L = \{ a^n b^n \mid n \geq 0 \}$ with $S_1 \to \lambda$, $S_2 \to aS_2b$, $S_2 \to ab$.
  - S_1: $L \oslash \lambda = \{ \Box, a\Box b, ab\Box, \Box ab, \dots \} = \{ u\Box v \mid uv \in L \}$
  - S_2: $L \oslash ab = L \oslash aabb = \{ \Box, a\Box b, aa\Box bb, \dots \} = \{ a^n\Box b^n \mid n \in \mathbb{N} \}$
- Dyck grammar: $S \to SS$, $S \to aSb$, $S \to \lambda$.
- Every regular language is generated by a congruential CFG

Congruential CFGs [Clark 2010]

Congruential context-free grammars

For every nonterminal N of G, if $u, v \in \mathcal{L}(G, N)$, then $u \equiv_{\mathcal{L}(G)} v$.

- If G is congruential, and we binarize G, then the result is congruential.
- So we assume productions are all like $N \to PQ$ or $N \to a$.

Examples not generated by congruential CFGs

- Palindromes: $\{ w \mid w = w^R \}$.
- $\{ a^n b^n \mid n \geq 0 \} \cup \{ a^n b^{2n} \mid n \geq 0 \}$
- $\{ a^m b^n \mid m \leq n \}$

Clark 2010 (modified)

Congruential CFGs are uniformly identifiable in the limit from positive data and membership queries.

Clark 2010 (modified)

Congruential CFGs are uniformly identifiable in the limit from positive data and membership queries.

Identification in the limit

- Input: Infinite sequence of the elements $w_1, w_2, \ldots$ of the learning target L_* in an arbitrary order
- Each time the learner gets an example w_i, it outputs a grammar G_i as her conjecture. After some point the conjecture should be stable and represent the target.

Membership Queries (MQs)

- Q: $w \in \Sigma^*$?
- A: Yes (if $w \in L_*$) ;
  No (otherwise).

Grammar Construction

$D \subseteq \Sigma^*$: finite set of examples.

Grammar Construction

$D \subseteq \Sigma^*$: finite set of examples.

The learner's hypothesis $G_{K,F}$ is computed from two sets

- $K \subseteq \mathrm{Sub}(D)$, where $\mathrm{Sub}(D) = \{\, u \mid \exists l, r.\ lur \in D \,\}$,
- $F \subseteq \mathrm{Con}(D)$, where $\mathrm{Con}(D) = \{\, l\Box r \mid \exists u.\ lur \in D \,\}$.

Grammar Construction

$D \subseteq \Sigma^*$: finite set of examples.
The learner's hypothesis $G_{K,F}$ is computed from two sets

- $K \subseteq \mathrm{Sub}(D)$, where $\mathrm{Sub}(D) = \{\, u \mid \exists l, r.\ lur \in D \,\}$,
- $F \subseteq \mathrm{Con}(D)$, where $\mathrm{Con}(D) = \{\, l\Box r \mid \exists u.\ lur \in D \,\}$.

$G_{K,F} = (\Sigma, V_K, I_K, R_K \cup R_{K,F})$ where

$$V_K = \{\, [\![u]\!] \mid u \in K \,\}.$$

We want $[\![u]\!]$ to generate all and only v s.t. $v \equiv_{L_*} u$,
i.e., $\mathcal{L}(G_{K,F}, [\![u]\!]) = [u]$.

Congruential context-free grammars

If $u, v \in \mathcal{L}(G, N)$, then $u \equiv_{\mathcal{L}(G)} v$.

Grammar Construction

$D \subseteq \Sigma^*$: finite set of examples.
The learner's hypothesis $G_{K,F}$ is computed from two sets

- $K \subseteq \mathrm{Sub}(D)$, where $\mathrm{Sub}(D) = \{\, u \mid \exists l, r.\ lur \in D \,\}$,
- $F \subseteq \mathrm{Con}(D)$, where $\mathrm{Con}(D) = \{\, l\Box r \mid \exists u.\ lur \in D \,\}$.

$G_{K,F} = (\Sigma, V_K, I_K, R_K \cup R_{K,F})$ where

- $V_K = \{\, [\![u]\!] \mid u \in K \,\}$,
- $I_K = \{\, [\![u]\!] \mid u \in L_* \,\}$ (by MQ),
- $R_K = \{\, [\![a]\!] \to a \mid a \in \Sigma \cup \{\lambda\} \,\} \cup \{\, [\![uv]\!] \to [\![u]\!][\![v]\!] \mid uv \in K \,\}$,
- $R_{K,F} = \{\, [\![u]\!] \to [\![v]\!] \mid (L_* \oslash u) \cap F = (L_* \oslash v) \cap F \,\}$,

with the aid of *Membership Queries*.

Monotonicity

Hypothesis grammar: $G_{K,F}$

- $V_K = \{ [\![u]\!] \mid u \in K \}$,
- $I_K = \{ [\![u]\!] \mid u \in L_* \}$ (by MQ),
- $R_K = \{ [\![a]\!] \to a \mid a \in \Sigma \cup \{\lambda\} \} \cup \{ [\![uv]\!] \to [\![u]\!][\![v]\!] \mid uv \in K \}$,
- $R_{K,F} = \{ [\![u]\!] \to [\![v]\!] \mid (L_* \oslash u) \cap F = (L_* \oslash v) \cap F \}$.

Monotonicity

If $K \subseteq K'$ then every rule of $G_{K,F}$ is a rule of $G_{K',F}$,
so $\mathcal{L}(G_{K,F}) \subseteq \mathcal{L}(G_{K',F})$.

Anti-Monotonicity

If $F \subseteq F'$ then every rule of $G_{K,F'}$ is a rule of $G_{K,F}$,
so $\mathcal{L}(G_{K,F}) \supseteq \mathcal{L}(G_{K,F'})$.

Chain Rule

$$R_{K,F} = \{ [\![u]\!] \to [\![v]\!] \mid (L_* \oslash u) \cap F = (L_* \oslash v) \cap F \}$$

	$\Box$	$a\Box$	$\Box b$
λ	1	0	0
a	0	0	1
b	0	1	0
ab	1	0	0
aab	0	0	1
abb	0	1	0

We have
$[\![a]\!] \leftrightarrow [\![aab]\!] \in R_{K,F}$
as they *look* congruent.

Anti-Monotonicity

If $F \subseteq F'$ then $R_{K,F} \supseteq R_{K,F'}$, and thus $\mathcal{L}(G_{K,F}) \supseteq \mathcal{L}(G_{K,F'})$.

Chain Rule

$$R_{K,F} = \{ [\![u]\!] \to [\![v]\!] \mid (L_* \oslash u) \cap F = (L_* \oslash v) \cap F \}$$

	$\square$	$a\square$	$\square b$	$\square abb$
λ	1	0	0	0
a	0	0	1	1
b	0	1	0	0
ab	1	0	0	0
aab	0	0	1	0
abb	0	1	0	0

NO $[\![a]\!] \leftrightarrow [\![aab]\!]$ any more.

Anti-Monotonicity

If $F \subseteq F'$ then $R_{K,F} \supseteq R_{K,F'}$, and thus $\mathcal{L}(G_{K,F}) \supseteq \mathcal{L}(G_{K,F'})$.

Example

	$\square$	$a\square$	$\square b$	$\square ab$	$\square abb$	$aab\square$
λ	1	0	0	1	0	0
a	0	0	1	0	1	0
b	0	1	0	0	0	1
ab	1	0	0	0	0	0
aab	0	0	1	0	0	0
abb	0	1	0	0	0	0
$aabb$	1	0	0	0	0	0

$$[\![aabb]\!] \in I_K$$

$$[\![aabb]\!] \Rightarrow [\![a]\!][\![abb]\!] \Rightarrow [\![a]\!][\![ab]\!][\![b]\!] \Rightarrow a[\![ab]\!]b \Rightarrow a[\![aabb]\!]b \Rightarrow aaabbb$$

$$aaabb \in \mathcal{L}(G_{K,F}),\ \text{in fact } \mathcal{L}(G_{K,F}) = \{\, a^n b^n \mid n \geq 0 \,\}$$

Soundness

We have $[\![u]\!] \to [\![v]\!]$ if $(L_* \oslash u) \cap F = (L_* \oslash v) \cap F$.

Soundness

We have $[\![u]\!] \to [\![v]\!]$ if $(L_* \oslash u) \cap F = (L_* \oslash v) \cap F$.
We say $[\![u]\!] \to [\![v]\!]$ is *incorrect* iff $L_* \oslash u \neq L_* \oslash v$, i.e., $[u] \neq [v]$.

Soundness

We have $[\![u]\!] \to [\![v]\!]$ if $(L_* \oslash u) \cap F = (L_* \oslash v) \cap F$.
We say $[\![u]\!] \to [\![v]\!]$ is *incorrect* iff $L_* \oslash u \neq L_* \oslash v$, i.e., $[u] \neq [v]$.

Every K admits finite F s.t. $G_{K,F}$ has no incorrect rules.

Proof. For each $u, v \in K$, if $L_* \oslash u \neq L_* \oslash v$, then there is $l\Box r \in (L_* \oslash u) \triangle (L_* \oslash v)$. Put $l\Box r$ into F.

Soundness

We have $[\![u]\!] \to [\![v]\!]$ if $(L_* \oslash u) \cap F = (L_* \oslash v) \cap F$.
We say $[\![u]\!] \to [\![v]\!]$ is *incorrect* iff $L_* \oslash u \neq L_* \oslash v$, i.e., $[u] \neq [v]$.

Every K admits finite F s.t. $G_{K,F}$ has no incorrect rules.

Proof. For each $u, v \in K$, if $L_* \oslash u \neq L_* \oslash v$, then there is $l\Box r \in (L_* \oslash u) \triangle (L_* \oslash v)$. Put $l\Box r$ into F.

If $G_{K,F}$ has no incorrect rules, $\mathcal{L}(G_{K,F}) \subseteq L_*$.

Proof.

- $[\![a]\!] \to a \quad \cdots \quad a \in [a]$,
- $[\![uv]\!] \to [\![u]\!][\![v]\!] \quad \cdots \quad [uv] \supseteq [u][v]$,
- $[\![u]\!] \to [\![v]\!] \quad \cdots \quad [u] = [v]$ since the rule is correct,

Hence $[\![u]\!] \overset{*}{\Rightarrow} v$ implies $v \in [u]$.

Soundness

We have $[\![u]\!] \to [\![v]\!]$ if $(L_* \oslash u) \cap F = (L_* \oslash v) \cap F$.
We say $[\![u]\!] \to [\![v]\!]$ is *incorrect* iff $L_* \oslash u \neq L_* \oslash v$, i.e., $[u] \neq [v]$.

Every K admits finite F s.t. $G_{K,F}$ has no incorrect rules.

Proof. For each $u, v \in K$, if $L_* \oslash u \neq L_* \oslash v$, then there is $l\Box r \in (L_* \oslash u) \triangle (L_* \oslash v)$. Put $l\Box r$ into F.

If $G_{K,F}$ has no incorrect rules, $\mathcal{L}(G_{K,F}) \subseteq L_*$.

Proof.

- $[\![a]\!] \to a \quad \cdots \quad a \in [a]$,
- $[\![uv]\!] \to [\![u]\!][\![v]\!] \quad \cdots \quad [uv] \supseteq [u][v]$,
- $[\![u]\!] \to [\![v]\!] \quad \cdots \quad [u] = [v]$ since the rule is correct,

Hence $[\![u]\!] \overset{*}{\Rightarrow} v$ implies $v \in [u]$.
Particularly for $[\![u]\!] \in I_K$, we have $v \in [u] \subseteq L_*$.

Completeness

Suppose L_* is generated by a congruential CFG G_*.

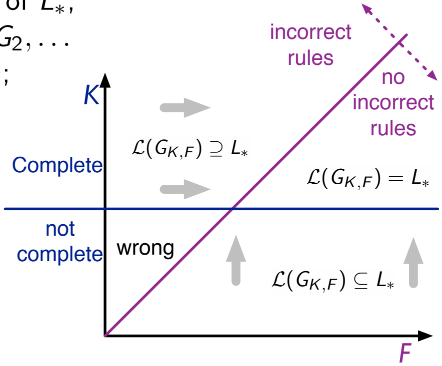
If $K \cap \mathcal{L}(G, \alpha) \neq \emptyset$ for every rule $N \to \alpha$ of G_*, then $L_* \subseteq \mathcal{L}(G_{K,F})$.

Proof. For a rule $N \to PQ$ of G_*, let $v_N, v_P, v_Q \in K$ be the shortest in $\mathcal{L}(G_*, N), \mathcal{L}(G_*, P), \mathcal{L}(G_*, Q)$, resp. Moreover we have $u_P u_Q \in K \cap \mathcal{L}(G_*, PQ)$.
$G_{K,F}$ has $[\![v_N]\!] \Rightarrow [\![v_P]\!][\![v_Q]\!]$ since

$$\begin{aligned}
[\![v_N]\!] &\to [\![u_P u_Q]\!] \text{ by } [v_N] = [u_P u_Q], \\
[\![u_P u_Q]\!] &\to [\![u_P]\!][\![u_Q]\!], \\
[\![u_P]\!] &\to [\![v_P]\!] \text{ by } [u_P] = [v_P], \\
[\![u_Q]\!] &\to [\![v_Q]\!] \text{ by } [u_Q] = [v_Q].
\end{aligned}$$

Learning Algorithm

Data: Positive data $w_1, w_2, \dots$ of L_*;
Result: Sequence of CFGs $G_1, G_2, \dots$
let $K := \emptyset$; $F := \emptyset$; $\hat{G} := G_{K,F}$;
for $n = 1, 2, \dots$ **do**
  let $D := \{w_1, \dots, w_n\}$;
  let $F := \mathrm{Con}(D)$;
  if $D \not\subseteq \mathcal{L}(\hat{G})$ **then**
    let $K := \mathrm{Sub}(D)$;
  end if
  output $\hat{G} = G_{K,F}$ as G_n;
end for



Distributional Learning

- Observe, model, exploit the relation between "substrings" and "contexts"
- [Primal] Learner for congruential CFGs uses Strings for nonterminals and Contexts for removing incorrect rules,
- [Dual] Use contexts for nonterminals and strings for removing incorrect rules.

Distributional Learning

- Observe, model, exploit the relation between "substrings" and "contexts"
- [Primal] Learner for congruential CFGs uses Strings for nonterminals and Contexts for removing incorrect rules,
- [Dual] Use contexts for nonterminals and strings for removing incorrect rules.
- Further generalization: each nonterminal is represented by sets rather than a single object.

	Primal	Dual
Nonterminal	string / set of strings	context / set of contexts
Rule validation	contexts	strings

Outline

Congruence on Sets

We have defined ...

- $l\Box r \odot u = lur$,
- $L_* \oslash u = \{\, l\Box r \mid (l\Box r) \odot u \subseteq L_* \,\}$,
- $L_* \oslash l\Box r = \{\, u \mid (l\Box r) \odot u \subseteq L_* \,\}$.
- $u \equiv_{L_*} v$ iff $L_* \oslash u = L_* \oslash v$.

Congruence on Sets

We have defined ...

- $l\Box r \odot u = lur$,
- $L_* \oslash u = \{\, l\Box r \mid (l\Box r) \odot u \subseteq L_* \,\}$,
- $L_* \oslash l\Box r = \{\, u \mid (l\Box r) \odot u \subseteq L_* \,\}$.
- $u \equiv_{L_*} v$ iff $L_* \oslash u = L_* \oslash v$.

For string set S and context set C, define

- $C \odot S = \{\, lur \mid l\Box r \in C \text{ and } u \in S \,\}$,
- $L_* \oslash S = \{\, l\Box r \mid (l\Box r) \odot S \subseteq L_* \,\}$,
- $L_* \oslash C = \{\, u \mid C \odot u \subseteq L_* \,\}$,
- $S \equiv_{L_*} T$ iff $L_* \oslash S = L_* \oslash T$.

Example

$L_* = \{ a^n b^n \mid n \geq 0 \}$.

	$\Box$	$a\Box$	$\Box b$	$a\Box b$	$a\Box bb$	$\Box abb$
λ	1	0	0	1	0	0
a	0	0	1	0	1	1
b	0	1	0	0	0	0
ab	1	0	0	1	0	0
aab	0	0	1	0	1	0
$aaabb$	0	0	1	0	1	0

- $L_* \oslash a = \{ \Box b, a\Box bb, \Box abb, \dots, a^i \Box a^j b^{i+j+1}, \dots \}$,
- $L_* \oslash aab = L_* \oslash \{aab, a\} = \{ \Box b, a\Box bb, \dots, a^k \Box b^{k+1}, \dots \}$,
- $\{aab\} \equiv_{L_*} \{a, aab, aaabb\} \not\equiv_{L_*} \{a\}$.

Learning Target

k-Kernel Property (Yoshinaka 2011)

A CFG G has *the k-KP* iff
every nonterminal N admits a finite set $S_N \subseteq \Sigma^*$ such that

- $|S_N| \leq k$,
- $S_N \equiv_{\mathcal{L}(G)} \mathcal{L}(G, N)$.

(Every congruential CFG has the 1-KP but not vice versa.)

Learning Target

k-Kernel Property (Yoshinaka 2011)

A CFG G has *the k-KP* iff
every nonterminal N admits a finite set $S_N \subseteq \Sigma^*$ such that

- $|S_N| \leq k$,
- $S_N \equiv_{\mathcal{L}(G)} \mathcal{L}(G, N)$.

(Every congruential CFG has the 1-KP but not vice versa.)

k-Context Property (Clark 2010)

A CFG G has *the k-CP* iff
every nonterminal N admits a finite set $C_N \subseteq \Sigma^*\Box\Sigma^*$ such that

- $|C_N| \leq k$,
- $\mathcal{L}(G) \oslash C_N = \mathcal{L}(G, N)$.

Learning Target

k-Kernel Property (Yoshinaka 2011)

A CFG G has *the k-KP* iff
every nonterminal N admits a finite set $S_N \subseteq \Sigma^*$ such that

- $|S_N| \leq k$,
- $S_N \equiv_{\mathcal{L}(G)} \mathcal{L}(G, N)$.

(Every congruential CFG has the 1-KP but not vice versa.)

k-Context Property (Clark 2010)

A CFG G has *the k-CP* iff
every nonterminal N admits a finite set $C_N \subseteq \Sigma^* \Box \Sigma^*$ such that

- $|C_N| \leq k$,
- $\mathcal{L}(G) \oslash C_N = \mathcal{L}(G, N)$.

Examples

- Every grammar G with a single nonterminal S has the 1-CP, since the initial symbol S is characterized by $C_S = \{\square\}$: $\mathcal{L}(G) \oslash \square = \mathcal{L}(G) = \mathcal{L}(G, S)$.
  - E.g. $\{ a^n b^n \mid n \geq 0 \}$,
    Palindrome $\{ w \in \Sigma^* \mid w = w^R \}$,
    Dyck language etc.

Examples

- Every grammar G with a single nonterminal S has the 1-CP, since the initial symbol S is characterized by $C_S = \{\square\}$: $\mathcal{L}(G) \oslash \square = \mathcal{L}(G) = \mathcal{L}(G, S)$.
    - E.g. $\{ a^n b^n \mid n \geq 0 \}$, Palindrome $\{ w \in \Sigma^* \mid w = w^R \}$, Dyck language etc.
- $\{a^n b^n \mid n \in \mathbb{N}\} \cup \{a^n b^{2n} \mid n \in \mathbb{N}\}$ has the 2-CP but not 1-CP.
    - $S_1 \to aS_1 b$, $S_1 \to \lambda$, $S_2 \to aS_2 bb$, $S_2 \to \lambda$.
    - $C_{S_1} = \{\square, a\square b\}$ and $C_{S_2} = \{\square, a\square bb\}$.
    - Note $\{a\square b\}$ does not characterize $\mathcal{L}(G, S_1)$ since $abbb \in (L \oslash a\square b) - \mathcal{L}(G, S_1)$.

Examples

- Every grammar G with a single nonterminal S has the 1-CP, since the initial symbol S is characterized by $C_S = \{\square\}$: $\mathcal{L}(G) \oslash \square = \mathcal{L}(G) = \mathcal{L}(G, S)$.
  - E.g. $\{ a^n b^n \mid n \geq 0 \}$,
    Palindrome $\{ w \in \Sigma^* \mid w = w^R \}$,
    Dyck language etc.
- $\{a^n b^n \mid n \in \mathbb{N}\} \cup \{a^n b^{2n} \mid n \in \mathbb{N}\}$ has the 2-CP but not 1-CP.
  - $S_1 \to aS_1b$, $S_1 \to \lambda$,
    $S_2 \to aS_2bb$, $S_2 \to \lambda$.
  - $C_{S_1} = \{\square, a\square b\}$ and $C_{S_2} = \{\square, a\square bb\}$.
  - Note $\{a\square b\}$ does not characterize $\mathcal{L}(G, S_1)$ since $abbb \in (L \oslash a\square b) - \mathcal{L}(G, S_1)$.
- Every regular language has the 1-CP.

Theorem

Theorem (Clark 2010, Yoshinaka 2011)

The class of CFGs with the k-CP is "efficiently" identifiable in the limit from positive data and MQs.

(k is known to the learner)

Grammar Construction

$D \subseteq L_*$: given set of positive examples.

- $F \subseteq \mathrm{Con}(D)$ and $K \subseteq \mathrm{Sub}(D)$.

$G_{F,K} = (\Sigma, V_F, I, R_F \cup R_{F,K})$ where

- $V_F = \{ [\![C]\!] \mid C \subseteq F \text{ and } |C| \leq k \}$,
  We want $\mathcal{L}(G_{F,K}, [\![C]\!]) = L_* \oslash C$

Grammar Construction

$D \subseteq L_*$: given set of positive examples.

- $F \subseteq \mathrm{Con}(D)$ and $K \subseteq \mathrm{Sub}(D)$.

$G_{F,K} = (\Sigma, V_F, I, R_F \cup R_{F,K})$ where

- $V_F = \{ [\![C]\!] \mid C \subseteq F \text{ and } |C| \leq k \}$,
  We want $\mathcal{L}(G_{F,K}, [\![C]\!]) = L_* \oslash C$
- $I = \{ [\![\{\Box\}]\!] \}$,
- $R_F = \{ [\![C]\!] \to a \mid a \in (L_* \oslash C) \cap (\Sigma \cup \{\lambda\}) \}$,
- $R_{F,K} = \{ [\![C_0]\!] \to [\![C_1]\!][\![C_2]\!] \mid (L_* \oslash C_0) \supseteq C_1^{(K)} C_2^{(K)} \}$,
  where $C^{(K)} = (L_* \oslash C) \cap K$.

Grammar Construction

$D \subseteq L_*$: given set of positive examples.

- $F \subseteq \mathrm{Con}(D)$ and $K \subseteq \mathrm{Sub}(D)$.

$G_{F,K} = (\Sigma, V_F, I, R_F \cup R_{F,K})$ where

- $V_F = \{ [\![C]\!] \mid C \subseteq F \text{ and } |C| \leq k \}$,
  We want $\mathcal{L}(G_{F,K}, [\![C]\!]) = L_* \oslash C$
- $I = \{ [\![\{\square\}]\!] \}$,
- $R_F = \{ [\![C]\!] \to a \mid a \in (L_* \oslash C) \cap (\Sigma \cup \{\lambda\}) \}$,
- $R_{F,K} = \{ [\![C_0]\!] \to [\![C_1]\!][\![C_2]\!] \mid (L_* \oslash C_0) \supseteq C_1^{(K)} C_2^{(K)} \}$,
  where $C^{(K)} = (L_* \oslash C) \cap K$.

We want $[\![C_0]\!] \to [\![C_1]\!][\![C_2]\!]$ iff $(L_* \oslash C_0) \supseteq (L_* \oslash C_1)(L_* \oslash C_2)$.

Monotonicity

If $F \subseteq F'$ then every rule of $G_{K,F}$ is a rule of $G_{K,F'}$,
so $\mathcal{L}(G_{K,F}) \subseteq \mathcal{L}(G_{K,F'})$.

Grammar Construction

$D \subseteq L_*$: given set of positive examples.

- $F \subseteq \mathrm{Con}(D)$ and $K \subseteq \mathrm{Sub}(D)$.

$G_{F,K} = (\Sigma, V_F, I, R_F \cup R_{F,K})$ where

- $V_F = \{ [\![C]\!] \mid C \subseteq F \text{ and } |C| \le k \}$,
  We want $\mathcal{L}(G_{F,K}, [\![C]\!]) = L_* \oslash C$
- $I = \{ [\![\{\square\}]\!] \}$,
- $R_F = \{ [\![C]\!] \to a \mid a \in (L_* \oslash C) \cap (\Sigma \cup \{\lambda\}) \}$,
- $R_{F,K} = \{ [\![C_0]\!] \to [\![C_1]\!][\![C_2]\!] \mid (L_* \oslash C_0) \supseteq C_1^{(K)} C_2^{(K)} \}$,
  where $C^{(K)} = (L_* \oslash C) \cap K$.

We want $[\![C_0]\!] \to [\![C_1]\!][\![C_2]\!]$ iff $(L_* \oslash C_0) \supseteq (L_* \oslash C_1)(L_* \oslash C_2)$.

Anti-Monotonicity

If $K \subseteq K'$ then every rule of $G_{K',F}$ is a rule of $G_{K,F}$,
so $\mathcal{L}(G_{K,F}) \supseteq \mathcal{L}(G_{K',F})$.

Correctness

We have $[\![C_0]\!] \to [\![C_1]\!][\![C_2]\!]$ if $C_0^{(\Sigma^*)} \supseteq C_1^{(K)} C_2^{(K)}$,
where $C^{(K)} = (L_* \oslash C) \cap K$.

We say that $[\![C_0]\!] \to [\![C_1]\!][\![C_2]\!]$ is *incorrect* iff $C_0^{(\Sigma^*)} \not\supseteq C_1^{(\Sigma^*)} C_2^{(\Sigma^*)}$.

Soundness Lemma

Every F admits finite K s.t. $G_{F,K}$ has no incorrect rules, in which case $\mathcal{L}(G_{F,K}) \subseteq L_*$.

Correctness

We have $[\![C_0]\!] \to [\![C_1]\!][\![C_2]\!]$ if $C_0^{(\Sigma^*)} \supseteq C_1^{(K)} C_2^{(K)}$,
where $C^{(K)} = (L_* \oslash C) \cap K$.

We say that $[\![C_0]\!] \to [\![C_1]\!][\![C_2]\!]$ is *incorrect* iff $C_0^{(\Sigma^*)} \not\supseteq C_1^{(\Sigma^*)} C_2^{(\Sigma^*)}$.

Soundness Lemma

Every F admits finite K s.t. $G_{F,K}$ has no incorrect rules, in which case $\mathcal{L}(G_{F,K}) \subseteq L_*$.

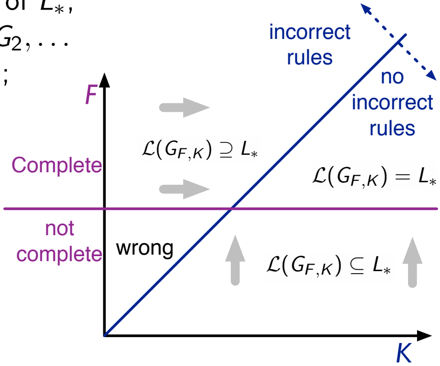
Suppose L_* is generated by a CFG G_* with the k-CP.
That is, every nonterminal N of G_* admits a context set C_N of cardinality at most k s.t. $L_* \oslash C_N \equiv_{L_*} \mathcal{L}(G_*, N)$.

Completeness Lemma

If $C_N \subseteq F$ for every N of G_*, then $L_* \subseteq \mathcal{L}(G_{F,K})$.

Learning Algorithm

Data: Positive data $w_1, w_2, \ldots$ of L_*;
Result: Sequence of CFGs $G_1, G_2, \ldots$
let $F := \varnothing$; $K := \varnothing$; $\hat{G} := G_{F,K}$;
for $n = 1, 2, \ldots$ **do**
  let $D := \{w_1, \ldots, w_n\}$;
  let $K := \mathrm{Sub}(D)$;
  if $D \not\subseteq \mathcal{L}(\hat{G})$ **then**
    let $F := \mathrm{Con}(D)$;
  end if
  output $\hat{G} = G_{F,K}$ as G_n;
end for



Theorems

Theorem (Clark 2010, Yoshinaka 2011)

The class of CFGs with the k-CP is "efficiently" identifiable in the limit from positive data and MQs.

Theorems

Theorem (Clark 2010, Yoshinaka 2011)

The class of CFGs with the k-CP is "efficiently" identifiable in the limit from positive data and MQs.

Theorem (Yoshinaka 2011)

The class of CFGs with the k-KP is "efficiently" identifiable in the limit from positive data and MQs.

Theorems

Theorem (Clark 2010, Yoshinaka 2011)

The class of CFGs with the k-CP is "efficiently" identifiable in the limit from positive data and MQs.

Theorem (Yoshinaka 2011)

The class of CFGs with the k-KP is "efficiently" identifiable in the limit from positive data and MQs.

Combined Property (Yoshinaka 2012)

Every nonterminal N admits either an m-kernel or n-context:

- a finite set $S_N \subseteq \Sigma^*$ s.t. $|S_N| \leq m$ and $S_N \equiv_{\mathcal{L}(G)} \mathcal{L}(G, N)$, or
- a finite set $C_N \subseteq \Sigma^* \square \Sigma^*$ s.t. $|C_N| \leq n$ and $\mathcal{L}(G) \oslash C_N \equiv_{\mathcal{L}(G)} \mathcal{L}(G, N)$

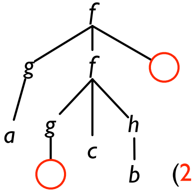
Outline

Simple Context-Free Tree Grammars

- Tree version of context-free (string) grammars
- (essentially) more general than Tree Adjoining (Substitution) Grammars
- (CFG) CF derivation trees yield strings
- (SCFTG) CF derivation trees yield trees

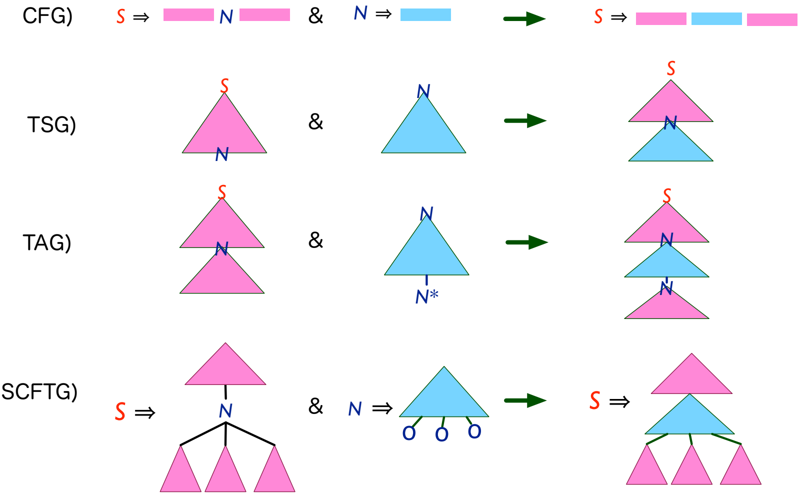
Trees and Stubs

- A ranked alphabet: $\Sigma = \bigcup_{0 \leq i \leq r} \Sigma_i$,
- If $t_1, \ldots, t_k$ are trees and $f \in \Sigma_k$ then $f(t_1, \ldots, t_k)$ is a tree,
- special symbol O of rank 0, which is a “hole”,
- a *k-stub* is a tree t over $\Sigma \cup \{O\}$ that contains exactly k holes, (0-stub = usual tree)
- each nonterminal of a CFG derives strings
- each nonterminal of rank k of an SCFTG derives k-stubs



(2-Stub)

Derivations of Different Formalisms



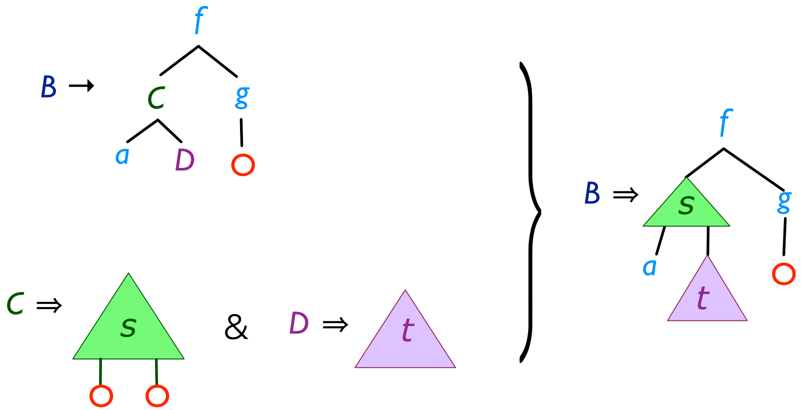
r-Simple Context-free Tree Grammar

A tuple $\langle \Sigma, V, I, R \rangle$ where

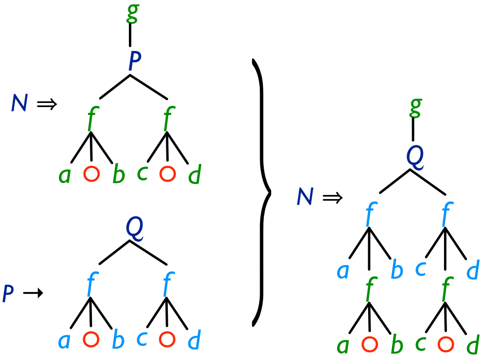
- Σ, V: finite set of ranked terminal/nonterminal symbols,
- $0 \leq \mathrm{rank}(a) \leq r$ for all $a \in \Sigma, V$,
- $I \subseteq V_0$: set of initial symbols (rank 0)
- $R \subseteq \bigcup_{k \leq r} V_k \times (k\text{-stubs})$: set of productions

$\mathcal{L}(G) = \bigcup_{S \in I} \mathcal{L}(G, S)$

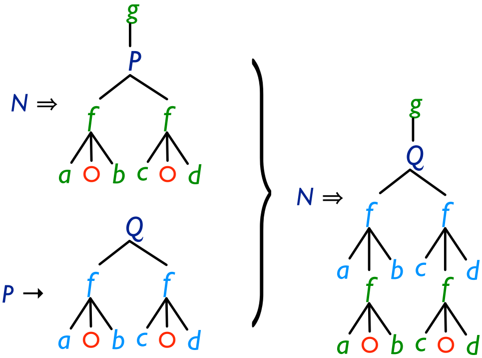
Derivation of SCFTG



Example



Example



Typical String languages of 2-SCFTGs:
$\{ a^n b^n c^n d^n \mid n \geq 1 \}$,
$\{ a^n b^n c^n d^n e^n f^n \mid n \geq 1 \}$ – not generated by a TAG

Substructure/Context Decomposition

CFGs

- $S \Rightarrow uAv$
- $A \Rightarrow w$
- $uwv \in L(G)$

  $u\square v$: **context**

  w : **substring**

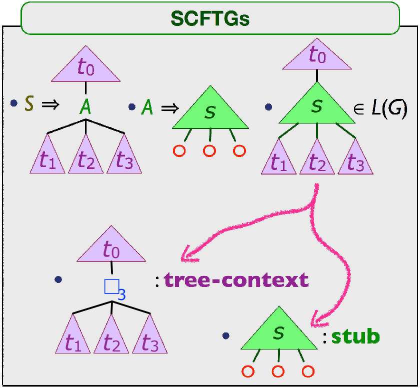
Substructure/Context Decomposition

CFGs

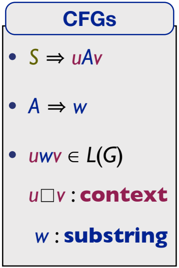
- $S \Rightarrow uAv$
- $A \Rightarrow w$
- $uwv \in L(G)$

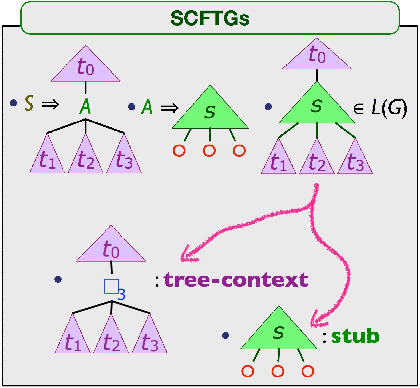
  $u\Box v$: **context**

  w : **substring**



Substructure/Context Decomposition





- k-tree context = tree with a hole $\square_k$ of rank k

Every technique on the distributional learning of CFGs can be translated and applied to the learning of SCFTGs!

Congruential SCFTGs

Congruential simple context-free tree grammars

An SCFTG G is said to be *congruential* if it satisfies the following: For any nonterminal N of rank k, if $s, t \in \mathcal{L}(G, N)$, then $s \equiv_{\mathcal{L}(G)} t$, that is, for any k-tree-context c, $c \odot s \in \mathcal{L}(G)$ iff $c \odot t \in \mathcal{L}(G)$.

Grammar Construction

$D \subseteq \Sigma^*$: finite set of tree examples

The learner's hypothesis G is computed from sets K_k, F_k with $0 \leq k \leq r$:

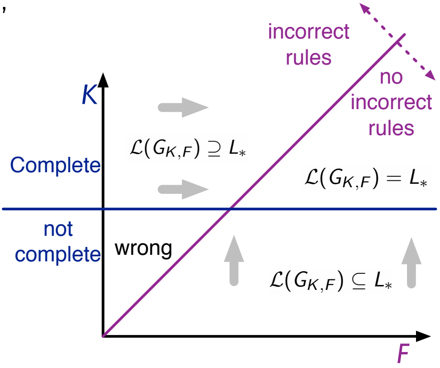
- $K_k \subseteq \mathrm{Sub}_k(D)$, where
  $\mathrm{Sub}_k(D)$ is the set of k-stubs extracted from D,
- $F_k \subseteq \mathrm{Con}_k(D)$, where
  $\mathrm{Con}_k(D)$ is the set of k-tree-contexts extracted from D.

$G = (\Sigma, V, I, R_K \cup R_{K,F})$ where

- $V = \bigcup_{k \leq r} V_k$ with $V_k = \{ [\![s]\!] \mid s \in K_k \}$,
- $I = \{ [\![t]\!] \mid t \in L_* \cap K_0 \}$ (by MQ),
- $R_K = \{ [\![f(\circ, \dots, \circ)]\!] \to f(\circ, \dots, \circ) \mid f \in \Sigma \}$
  $\cup \{ [\![s_0]\!] \to [\![s_1]\!](\circ, \dots, \circ, [\![s_2]\!](\circ, \dots, \circ), \circ, \dots, \circ)$
  $\mid s_0 = s_1(\circ, \dots, \circ, s_2(\circ, \dots, \circ), \circ, \dots, \circ) \}$,
- $R_{K,F} = \{ [\![s]\!] \to [\![t]\!] \mid c \odot s \in \mathcal{L}(G)$ iff $c \odot t \in \mathcal{L}(G)$ for all $c \in F \}$,

Learning Algorithm

Data: Positive data $t_1, t_2, \ldots$ of L_*;
Result: Sequence of SCFTGs $G_1, G_2, \ldots$
let $K := \varnothing$; $F := \varnothing$; $\hat{G} := G_{K,F}$;
for $n = 1, 2, \ldots$ **do**
  let $D := \{t_1, \ldots, t_n\}$;
  let $F_k := \mathrm{Con}_k(D)$
    for $k = 0, \ldots, r$;
  if $D \not\subseteq \mathcal{L}(\hat{G})$ **then**
    let $K_k := \mathrm{Sub}_k(D)$
      for $k = 0, \ldots, r$;
  end if
  output $\hat{G} = G_{K,F}$ as G_n;
end for



Other Properties and Learning

Similarly one can define k-KP and k-CP for SCFTGs
and design learning algorithms for the corresponding classes.

Other “context-free” formalisms

$$N \rightarrow \alpha[P_1, \dots, P_k]$$

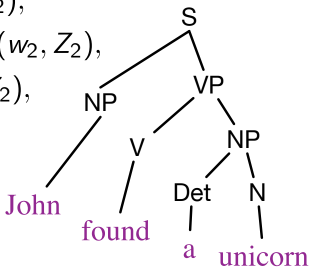
- Context-free grammars (Clark & Eyraud’07 etc....)
  - $N \rightarrow$ string
- Simple context-free tree grammars (Kasprzik & Yoshinaka ’11)
  - $N \rightarrow$ tree/stub
- Multiple CFGs (Yoshinaka ’12 etc.)
  - $N \rightarrow$ tuple of strings
- Linear context-fee linear λ-grammars (Yoshinaka & Kanazawa’11)
  - $N \rightarrow \lambda$-term

Context-free grammar with Montague semantics

$$S(w_1 w_2, Z_1 Z_2) \mathrel{:-} NP(w_1, Z_1) VP(w_2, Z_2),$$
$$VP(w_1 w_2, \lambda x. Z_2(\lambda y. Z_1 y x)) \mathrel{:-} V(w_1, Z_1) NP(w_2, Z_2),$$
$$NP(w_1 w_2, Z_1 Z_2) \mathrel{:-} Det(w_1, X_1) N(w_2, Z_2),$$
$$NP(\text{John}, \lambda u. u\, \textsc{John}) \mathrel{:-},$$
$$V(\text{found}, \lambda yz. \textsc{find}\, yz) \mathrel{:-},$$
$$Det(\text{a}, \lambda uv. \textsc{Intersect}\, uv) \mathrel{:-},$$
$$N(\text{unicorn}, \lambda y. \textsc{unicorn}\, y) \mathrel{:-}$$

Context-free grammar with Montague semantics

$$S(w_1 w_2, Z_1 Z_2) :\!-\; NP(w_1, Z_1) VP(w_2, Z_2),$$
$$VP(w_1 w_2, \lambda x. Z_2(\lambda y. Z_1 y x)) :\!-\; V(w_1, Z_1) NP(w_2, Z_2),$$
$$NP(w_1 w_2, Z_1 Z_2) :\!-\; Det(w_1, X_1) N(w_2, Z_2),$$
$$NP(\text{John}, \lambda u. u\ \textsc{John}) :\!-,$$
$$V(\text{found}, \lambda yz. \textsc{find}\ yz) :\!-,$$
$$Det(\text{a}, \lambda uv. \textsc{Intersect}\ uv) :\!-,$$
$$N(\text{unicorn}, \lambda y. \textsc{unicorn}\ y) :\!-$$



$$\langle \text{John found a unicorn}, \textsc{Intersect}(\lambda y. \textsc{unicorn}\ y)(\lambda y. \textsc{find}\ y\ \textsc{John})\rangle$$

Summary

Distributional Learning

- Context-substring relation
- Primal-dual approaches
- Correctness of rules
- Monotonicity with respect to the two sets

	Primal	Dual
Nonterminal	string / set of strings	context / set of contexts
Rule validation	contexts	strings

Other formalisms

- Simple context-free tree grammars (Kasprzik & Yoshinaka '11)
- Multiple CFGs (Yoshinaka '12 etc.)
- Linear context-fee linear λ-grammars (Yoshinaka & Kanazawa '11)