An Extended GHKM Algorithm for Inducing λ-SCFG

Peng Li, Yang Liu and Maosong Sun

THUNLP&CSS
Tsinghua University, China

Outline

- Background
- Rule extraction algorithm
- Modeling
- Experiments
- Conclusion

Outline

- Background
- Rule extraction algorithm
- Modeling
- Experiments
- Conclusion

Semantic Parsing

- Semantic parsing: mapping a natural language sentence into its computer executable meaning representation

NL : Every boy likes a star

MR: $\forall x.(\mathit{boy}\ (x) \rightarrow \exists y\ (\mathit{human}(y) \wedge \mathit{pop}(y) \wedge \mathit{like}(x, y)))$

Related Work

- Hand-build systems (e.g., Woods et al., 1972; Warren & Pereira, 1982)
- Learning for semantic parsing
  - Supervised methods (e.g., Wong & Mooney, 2007; Lu et al., 2008)
  - Semi-supervised methods (e.g., Kate & Mooney, 2007)
  - Unsupervised methods (e.g., Poon & Domingos, 2009 & 2010; Goldwasser et al., 2011)

Related Work

- Hand-build systems (e.g., Woods et al., 1972; Warren & Pereira, 1982)
- Learning for semantic parsing
  - Supervised methods (e.g., Wong & Mooney, 2007; Lu et al., 2008)
  - Semi-supervised methods (e.g., Kate & Mooney, 2007)
  - Unsupervised methods (e.g., Poon & Domingos, 2009 & 2010; Goldwasser et al., 2011)

Supervised Methods

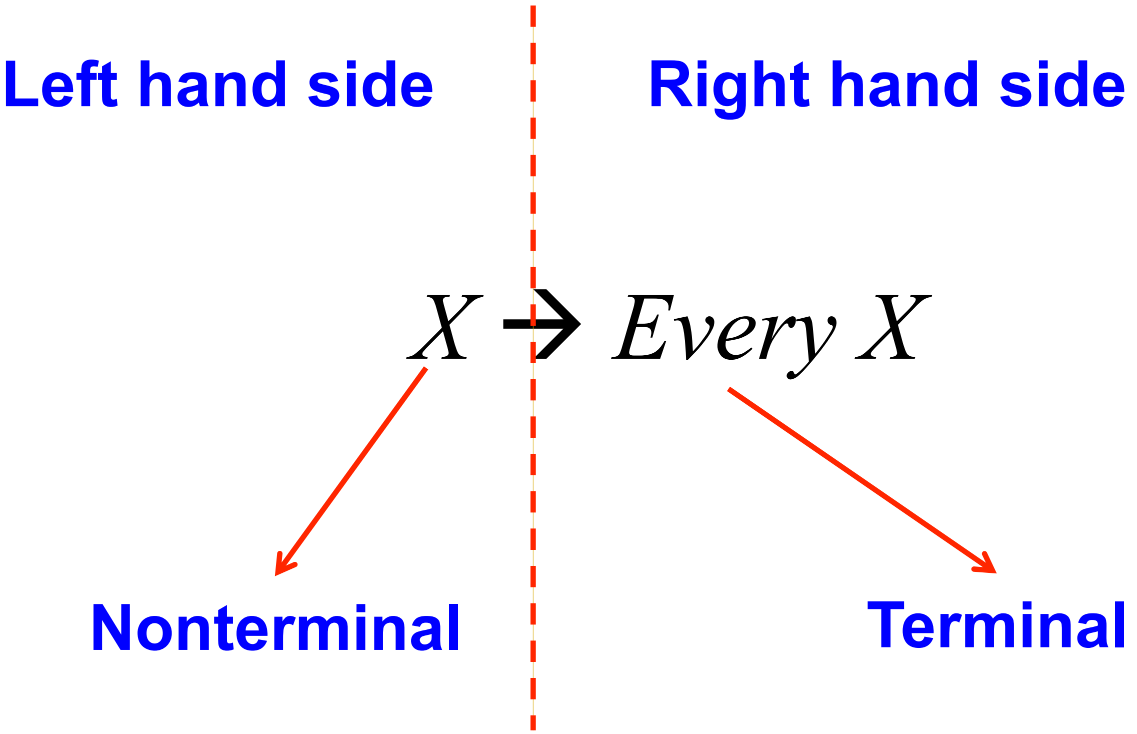
- Inductive logic programming based methods (e.g., Zelle & Mooney, 1996; Tang & Mooney, 2001)
- String kernel based methods (e.g., Kate & Mooney, 2006)
- Grammar based methods
  - PCFG (e.g., Ge & Mooney, 2005)
  - SCFG (e.g., Wong & Mooney, 2006 & 2007)
  - CCG (e.g., Zettlemoyer & Collins, 2005 & 2007; Kwiatkowski et al., 2010 & 2011)
  - Hybrid tree (e.g., Lu et al., 2008)
  - Tree transducer (Jones et al., 2012)

Supervised Methods

- Inductive logic programming based methods (e.g., Zelle & Mooney, 1996; Tang & Mooney, 2001)
- String kernel based methods (e.g., Kate & Mooney, 2006)
- Grammar based methods
  - PCFG (e.g., Ge & Mooney, 2005)
  - SCFG (e.g., Wong & Mooney, 2006 & 2007)
  - CCG (e.g., Zettlemoyer & Collins, 2005 & 2007; Kwiatkowski et al., 2010 & 2011)
  - Hybrid tree (e.g., Lu et al., 2008)
  - Tree transducer (Jones et al., 2012)

Context Free Grammar (CFG)

- A formal grammar in which every production rule is of the following form



Left hand side

Right hand side

$$X \rightarrow Every\ X$$

Nonterminal

Terminal

Context Free Grammar (CFG)

- Derivation example

Derivation

$S \rightarrow X$

$\rightarrow \textit{Every}\ X$

$\rightarrow \textit{Every}\ X_1\ X_2$

$\rightarrow \textit{Every boy}\ X_2$

$\rightarrow \textit{Every boy}\ X_2\ \textit{a star}$

$\rightarrow \textit{Every boy likes a star}$

CFG Rules

$r_1: S \rightarrow X$

$r_2: X \rightarrow \textit{Every}\ X$

$r_3: X \rightarrow X_1\ X_2$

$r_4: X \rightarrow \textit{boy}$

$r_5: X \rightarrow X\ \textit{a star}$

$r_6: X \rightarrow \textit{likes}$

Synchronous Context Free Grammar (SCFG)

Left hand side

Right hand side 1

Right hand side 2

$$X \rightarrow < Every\ \underbrace{X_1, \text{每个}\ X_1} >$$

Rewritten synchronously

One nonterminal

Synchronous Context Free Grammar (SCFG)

- Two strings can be generated synchronously

$S \rightarrow < X, X >$

$\rightarrow < Every\ X,$ 每个 $X >$

$\rightarrow < Every\ X_1\ X_2,$ 每个 $X_1\ X_2 >$

……….

$\rightarrow < Every\ boy\ likes\ a\ star,$

每个 男孩 都 喜欢 一个 明星>

How to use SCFG to handle logical forms?

λ-calculus

- A formal system in mathematical logic for expressing computation by way of variable binding and substitution
  - λ-expression: $\lambda x.\lambda y.borders(y, x)$
  - β-conversion: bound variable substitution

    $$\lambda x.\lambda y.borders(y, x)(texas) = \lambda y.borders(y, texas)$$
  - α-conversion: bound variable renaming

    $$\lambda x.\lambda y.borders(y, x) = \lambda z.\lambda y.borders(y, z)$$

λ-SCFG: SCFG+λ-calculus

- Reducing semantic parsing problem to SCFG parsing problem
- Using λ-calculus to handle semantic specific phenomenon
- Rule example
  - $X \rightarrow < Every\ X_1\ , \lambda f. \forall x\ (f\ (x)) \triangleleft X_1 >$

(Wong & Mooney, 2007)

λ-SCFG: SCFG+λ-calculus

NL : Every boy likes a star

r_1: $S \rightarrow < X_1, X_1 >$

r_2: $X \rightarrow < Every\ X_1, \lambda f.\forall x\ (f(x))\ \lhd X_1 >$

r_3: $X \rightarrow < X_1\ X_2, \lambda f.\ \lambda g.\lambda x.\ f(x) \rightarrow g(x)\ \lhd X_1 \lhd X_2 >$

r_4: $X \rightarrow < boy, \lambda x.boy\ (x) >$

r_5: $X \rightarrow < X_1, \lambda f.\ \lambda x.\ \exists y\ (f(x, y))\ \lhd X_1 >$

r_6: $X \rightarrow < X_1\ a\ star, \lambda f.\ \lambda x.\lambda y.\ human(y) \wedge pop(y) \wedge\ f(x,y) \lhd X_1 >$

r_7: $X \rightarrow < like, \lambda x.\lambda y.like(x,y) >$

$< S_1, S_1 >$

$\rightarrow < X_2, X_2 >$ (r_1)

$\rightarrow < Every\ X_3, \lambda f.\forall x.\ (f(x))\ \lhd X_3 >$ (r_2)

$\rightarrow < Every\ X_4\ X_5, \lambda f.\ \lambda g.\forall x.(f(x) \rightarrow g(x)) \lhd X_4 \lhd X_5 >$ (r_3)

$\rightarrow < Every\ boy\ X_5, \lambda g.\forall x.(boy\ (x) \rightarrow g(x)) \lhd X_5 >$ (r_4)

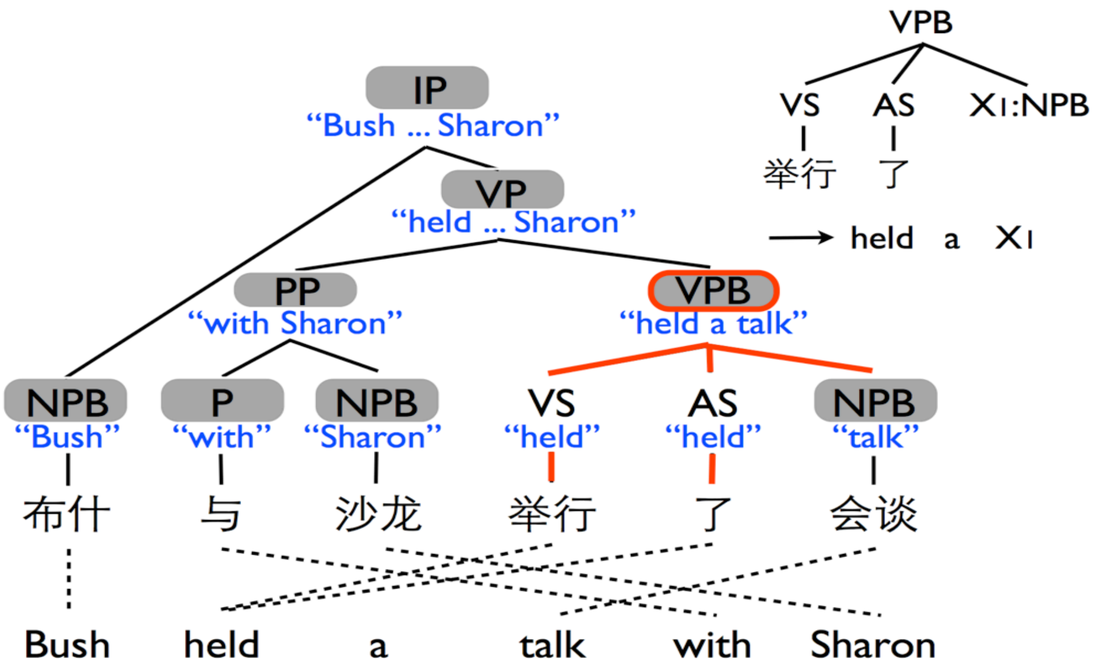
$\rightarrow < Every\ boy\ X_6, \lambda f.\forall x.(boy\ (x) \rightarrow \exists y(f(x, y))) \lhd X_6 >$ (r_5)

$\rightarrow < Every\ boy\ X_7\ a\ star, \lambda f.\forall x.(boy\ (x) \rightarrow \exists y\ (human(y) \wedge pop(y) \wedge f(x, y))) \lhd X_7 >$ (r_6)

$\rightarrow < Every\ boy\ likes\ a\ star, \forall x.(boy\ (x) \rightarrow \exists y\ (human(y) \wedge pop(y) \wedge like(x, y))) >$ (r_7)

GHKM

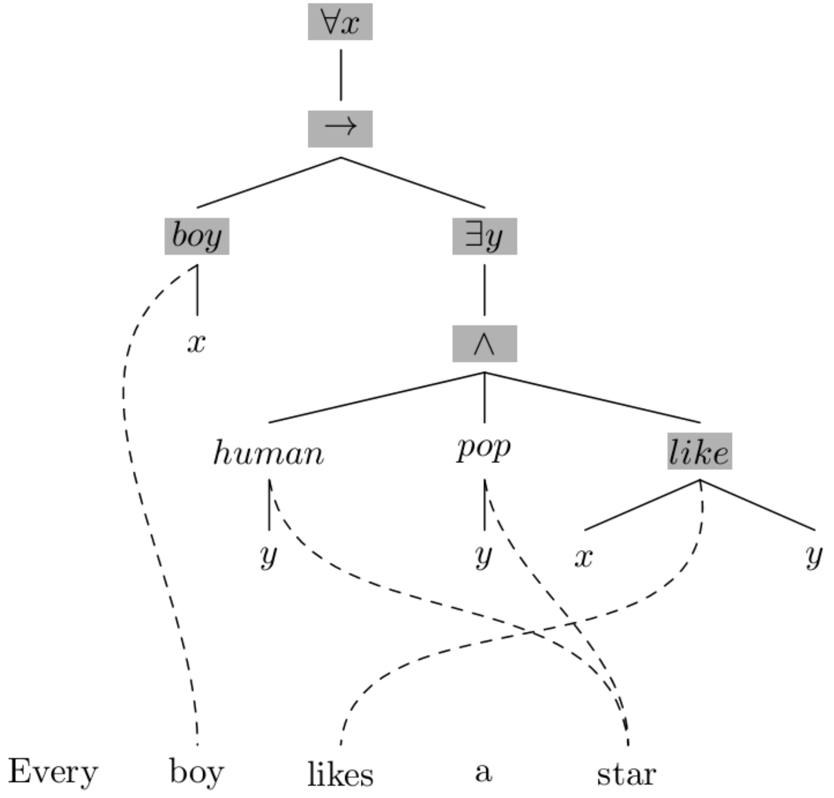
- The GHKM algorithm extracts STSG rules from aligned tree-string pairs



(Galley et al., 2004)

GHKM

- The GHKM algorithm extracts STSG rules from aligned tree-string pairs

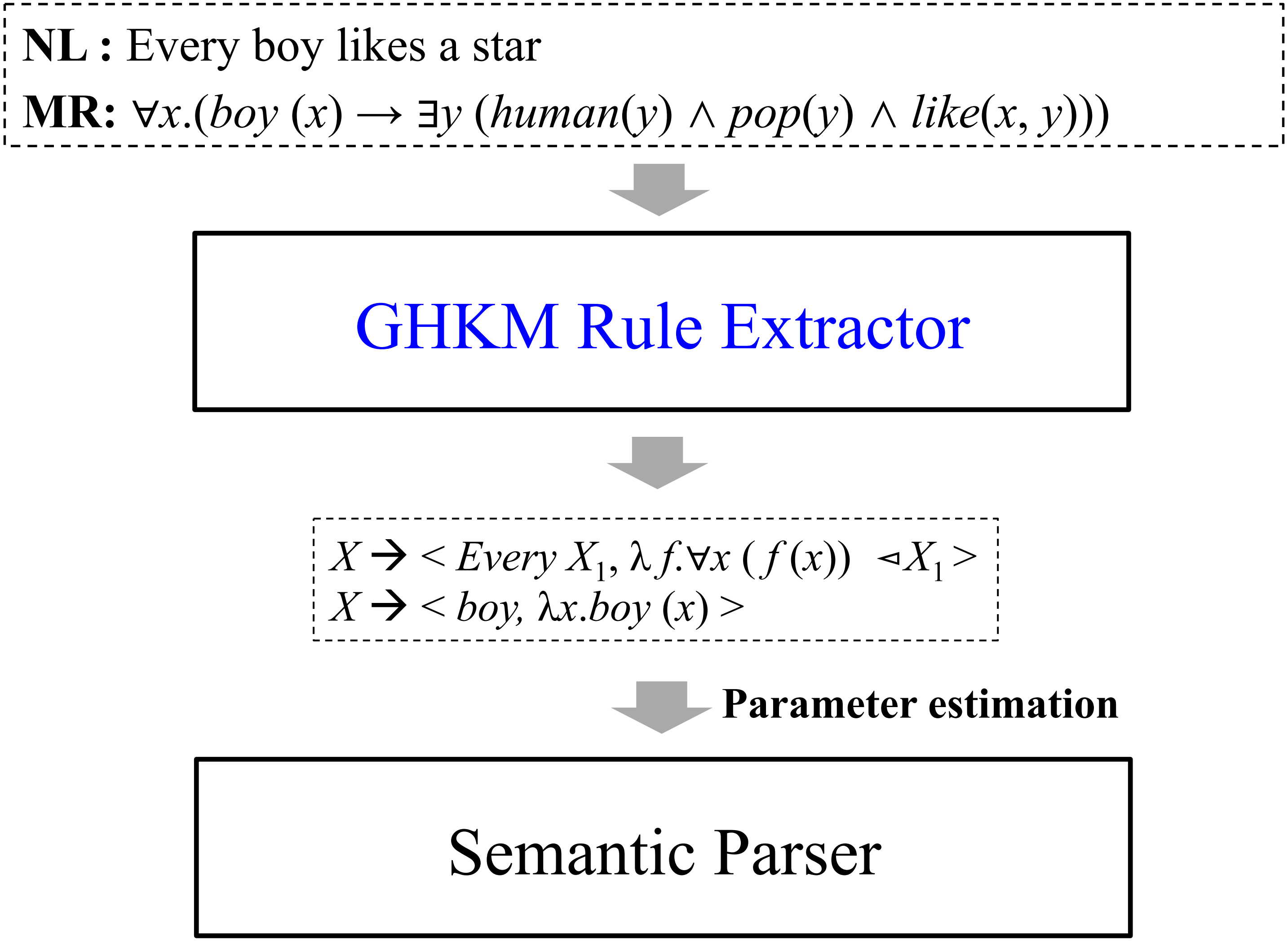


Our work

Outline

- Background
- Rule extraction algorithm
- Modeling
- Experiments
- Conclusion

Overview



NL : Every boy likes a star

MR: $\forall x.(boy\ (x) \rightarrow \exists y\ (human(y) \wedge pop(y) \wedge like(x, y)))$

GHKM Rule Extractor

$X \rightarrow <Every\ X_1, \lambda f.\forall x\ (f\ (x)) \lhd X_1>$

$X \rightarrow <boy, \lambda x.boy\ (x)>$

Parameter estimation

Semantic Parser

Rule Extraction Algorithm

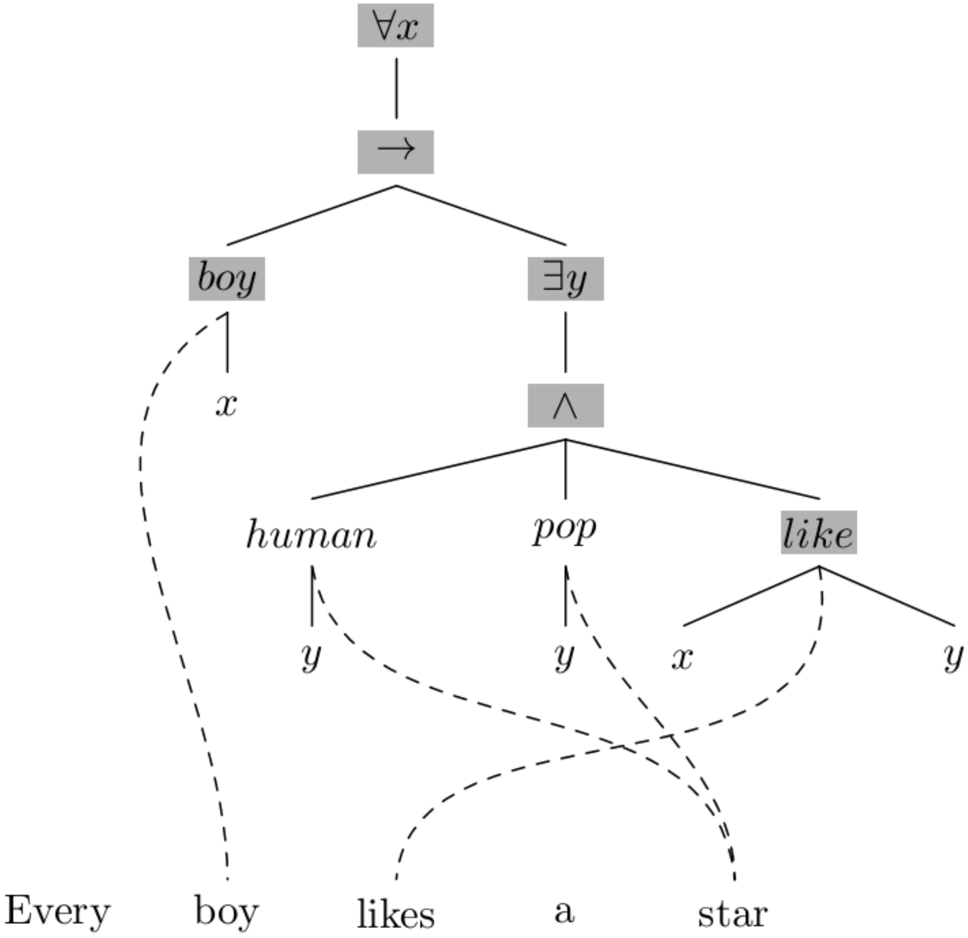
- Outline
  1. Building training examples
     1. Transforming logical forms to trees
     2. Aligning trees with sentences
  2. Identifying frontier nodes
  3. Extracting minimal rules
  4. Extracting composed rules

Building Training Examples

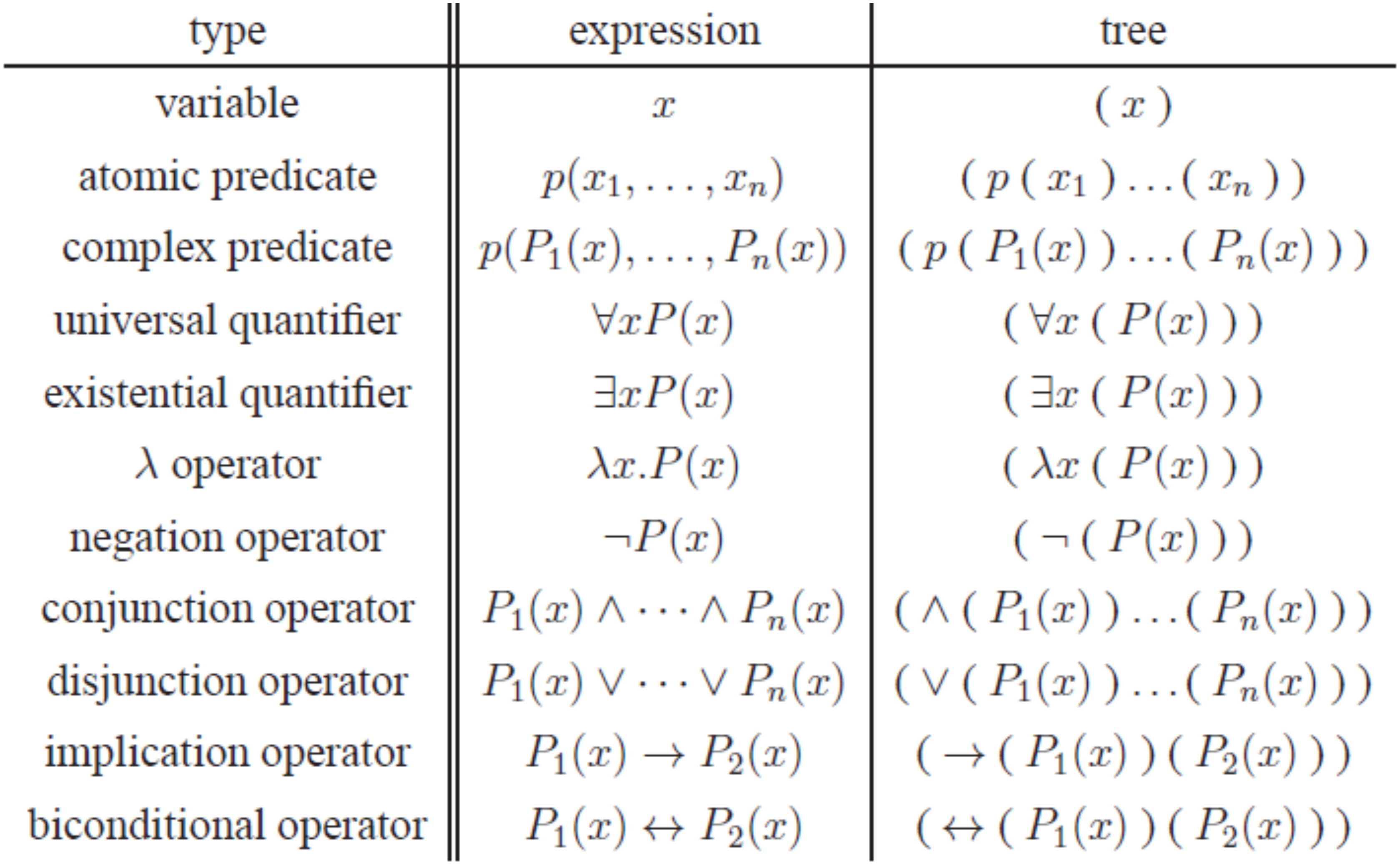
NL : Every boy likes a star

MR: $\forall x.(boy\ (x) \rightarrow \exists y\ (human(y) \wedge pop(y) \wedge like(x, y)))$

Building Training Examples



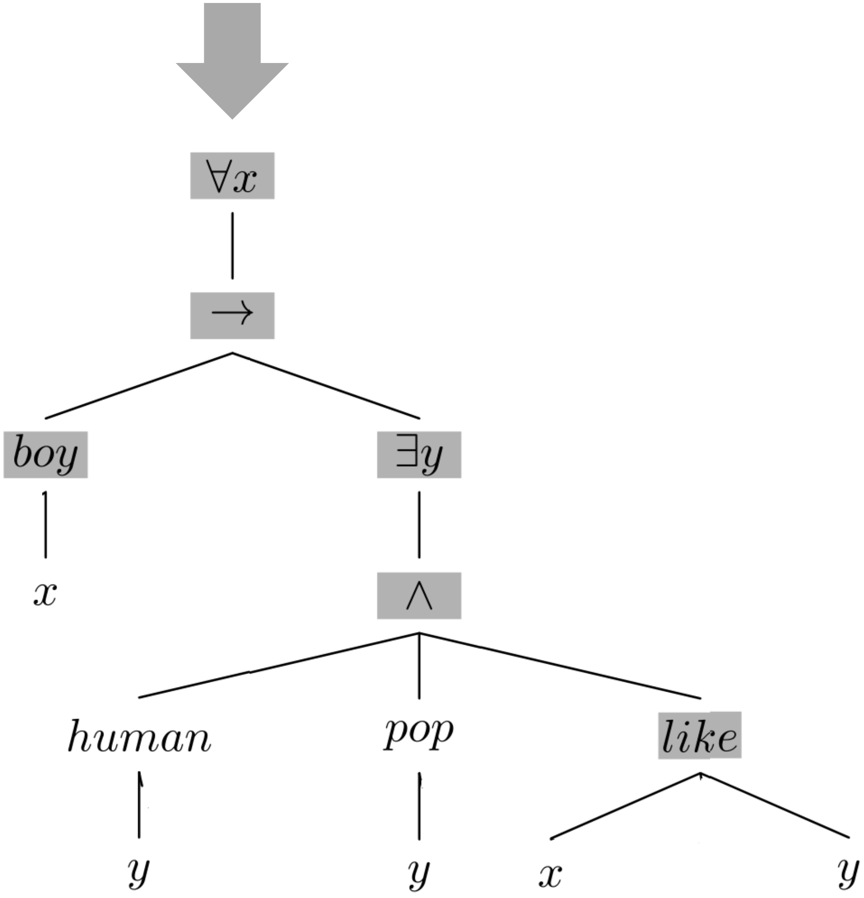
Building Training Examples



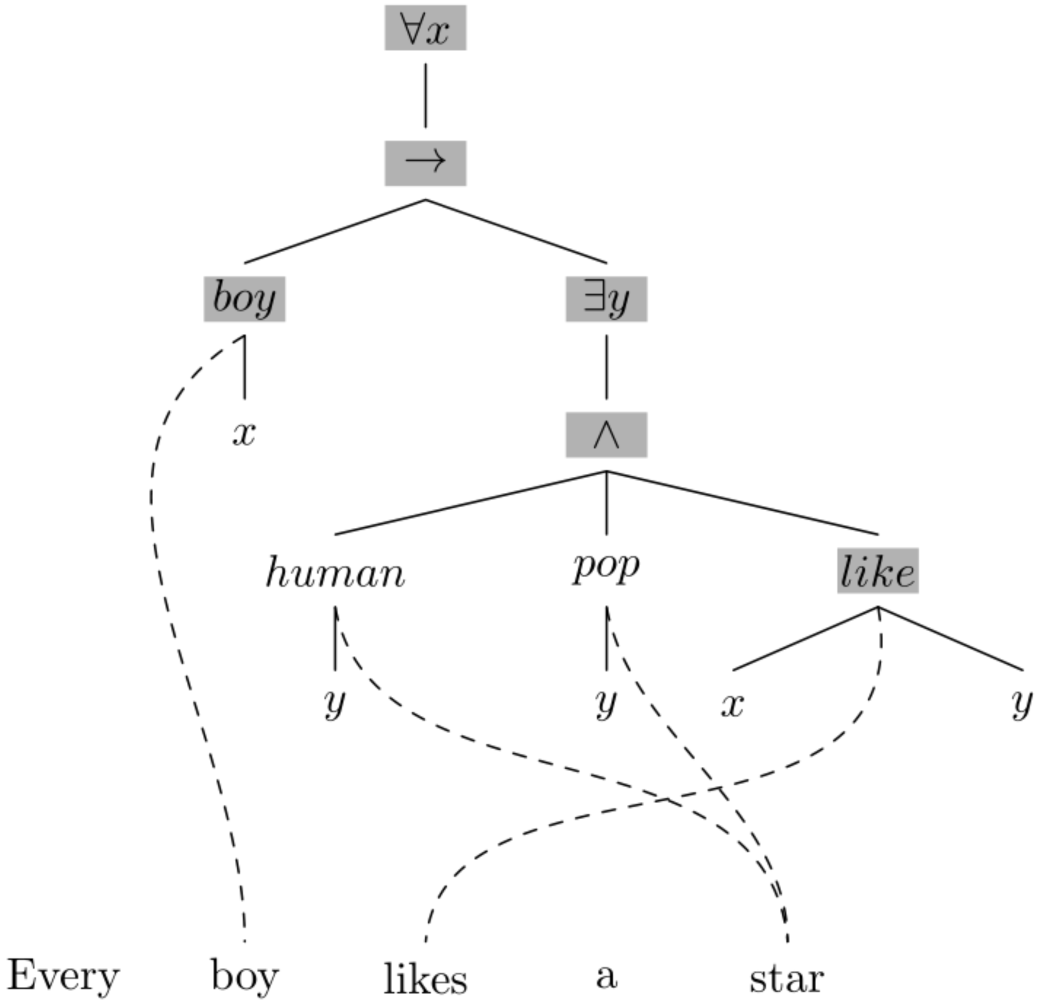
type	expression	tree
variable	x	$(\; x \;)$
atomic predicate	$p(x_1, \ldots, x_n)$	$(\; p \; (\; x_1 \;) \ldots (\; x_n \;) \;)$
complex predicate	$p(P_1(x), \ldots, P_n(x))$	$(\; p \; (\; P_1(x) \;) \ldots (\; P_n(x) \;) \;)$
universal quantifier	$\forall x P(x)$	$(\; \forall x \; (\; P(x) \;) \;)$
existential quantifier	$\exists x P(x)$	$(\; \exists x \; (\; P(x) \;) \;)$
λ operator	$\lambda x.P(x)$	$(\; \lambda x \; (\; P(x) \;) \;)$
negation operator	$\neg P(x)$	$(\; \neg \; (\; P(x) \;) \;)$
conjunction operator	$P_1(x) \wedge \cdots \wedge P_n(x)$	$(\; \wedge \; (\; P_1(x) \;) \ldots (\; P_n(x) \;) \;)$
disjunction operator	$P_1(x) \vee \cdots \vee P_n(x)$	$(\; \vee \; (\; P_1(x) \;) \ldots (\; P_n(x) \;) \;)$
implication operator	$P_1(x) \to P_2(x)$	$(\; \to \; (\; P_1(x) \;) \; (\; P_2(x) \;) \;)$
biconditional operator	$P_1(x) \leftrightarrow P_2(x)$	$(\; \leftrightarrow \; (\; P_1(x) \;) \; (\; P_2(x) \;) \;)$

Building Training Examples

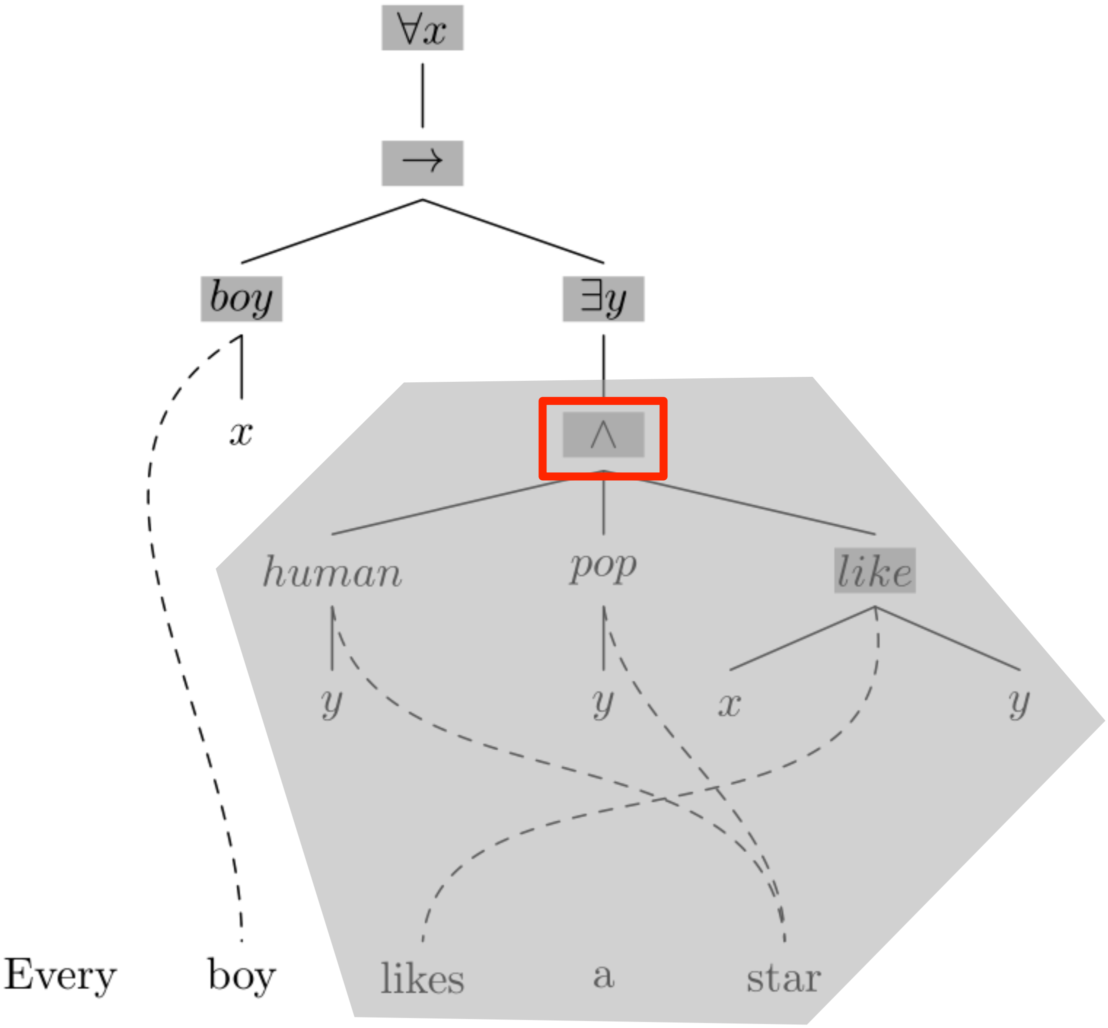
$$\forall x.(boy\ (x) \rightarrow \exists y\ (human(y) \wedge pop(y) \wedge like(x, y)))$$



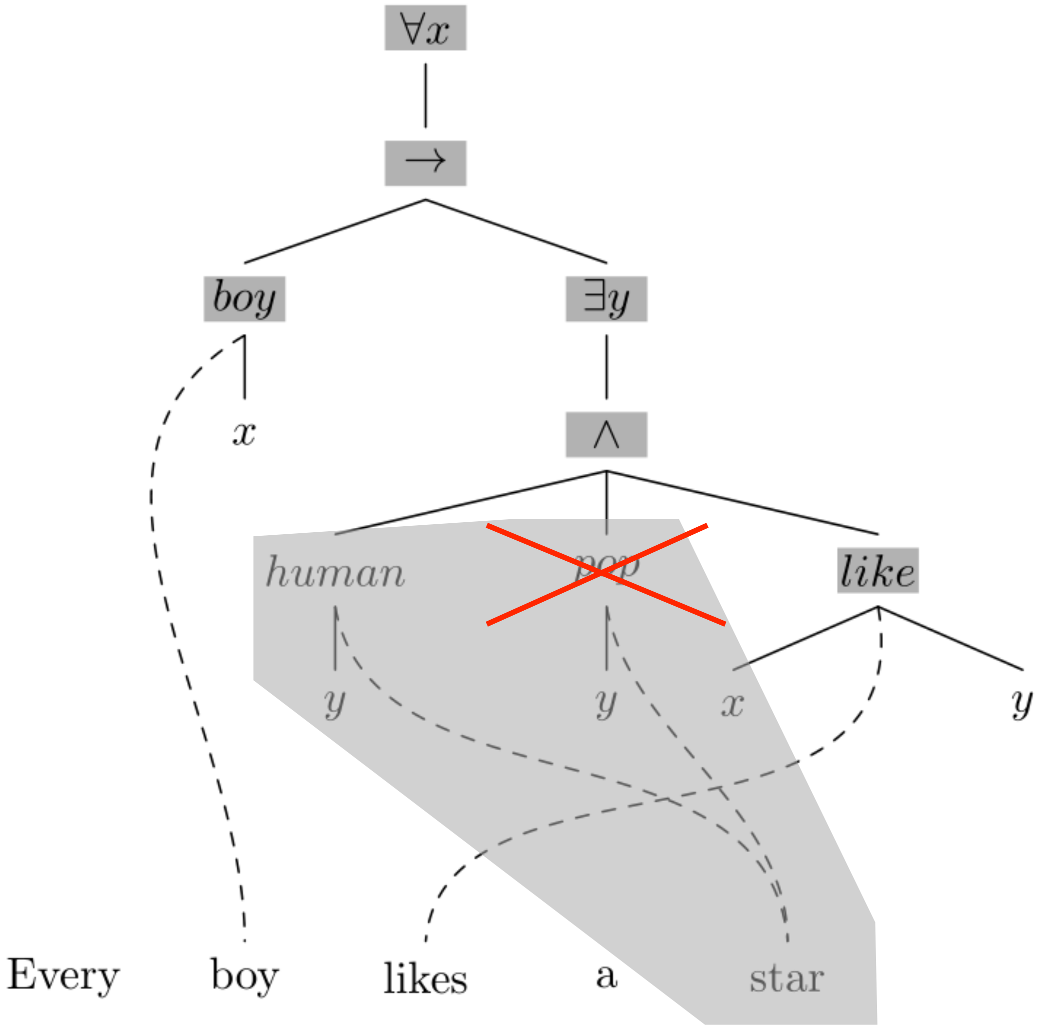
Building Training Examples



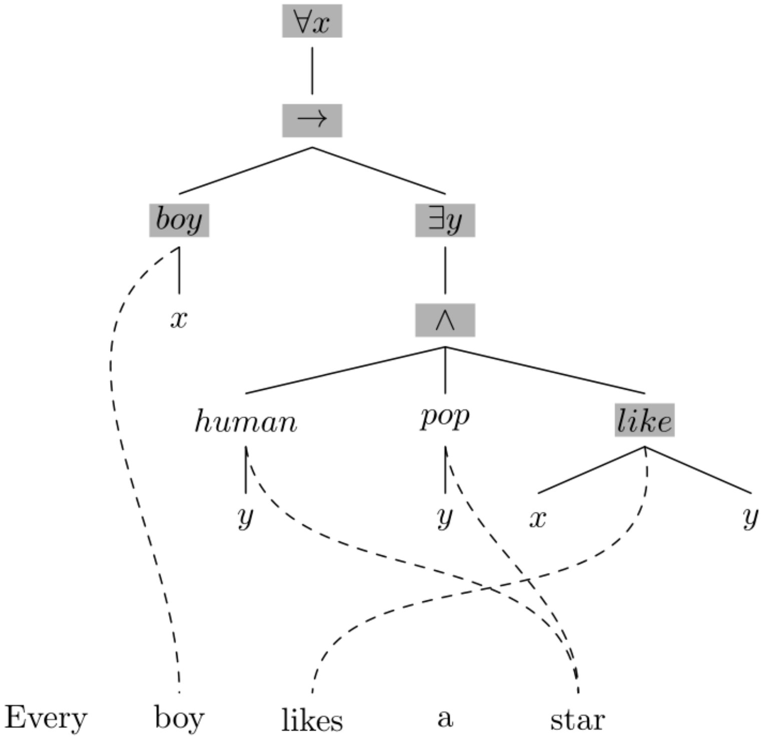
Identifying Frontier Nodes



Identifying Frontier Nodes



Extracting minimal rules



$\forall x$: $X \rightarrow <$ *Every* $X_1, \lambda f.\forall x\ (f(x)) \ \triangleleft X_1 >$

$\rightarrow$: $X \rightarrow < X_1\ X_2, \lambda f.\ \lambda g.\lambda x.\ f(x) \rightarrow g(x) \triangleleft X_1 \triangleleft X_2 >$

boy: $X \rightarrow <$ *boy*, $\lambda x.boy\ (x) >$

$\exists y$: $X \rightarrow < X_1, \lambda f.\ \lambda x.\ \exists y\ (f(x, y)) \triangleleft X_1 >$

$\wedge$: $X \rightarrow < X_1$ *a star*,
$\lambda f.\ \lambda x.\lambda y.\ human(y) \wedge pop(y) \wedge\ f(x,y) \triangleleft X_1 >$

like: $X \rightarrow <$ *like*, $\lambda x.\lambda y.like(x,y) >$

Composed Rule Extraction

$$X \rightarrow < X_1\ X_2,\ \lambda f.\ \lambda g.\lambda x.\ f(x) \rightarrow g(x)\ \lhd X_1 \lhd X_2 >$$

$$X \rightarrow < boy,\ \lambda x.boy\ (x) >$$

$$+\ X \rightarrow < X_1,\ \lambda f.\ \lambda x.\ \exists y\ (f\ (x,\ y))\ \lhd X_1 >$$

$$=\ X \rightarrow < boy\ X_1,\ \lambda f.\ \lambda x.boy(x) \rightarrow\ \exists y\ (f\ (x,\ y)) \lhd X_1 >$$

Outline

- Background
- Rule extraction algorithm
- Modeling
- Experiments
- Conclusion

Modeling

- Log-linear model + MERT training

$$w(D) = \prod_{i=1}^{3} \prod_{r \in D} h_i(r)^{\lambda_i} \times h_4(D)^{\lambda_4} \times h_5(D)^{\lambda_5}$$

$$h_1(X \rightarrow \langle s, e \rangle) = p(e \mid s) \qquad h_4(X \rightarrow \langle s, e \rangle) = p_s(e(D))$$

$$h_2(X \rightarrow \langle s, e \rangle) = p_{lex}(s \mid e) \qquad h_5(X \rightarrow \langle s, e \rangle) = \exp(|D|)$$

$$h_3(X \rightarrow \langle s, e \rangle) = p_{lex}(e \mid s)$$

- Target

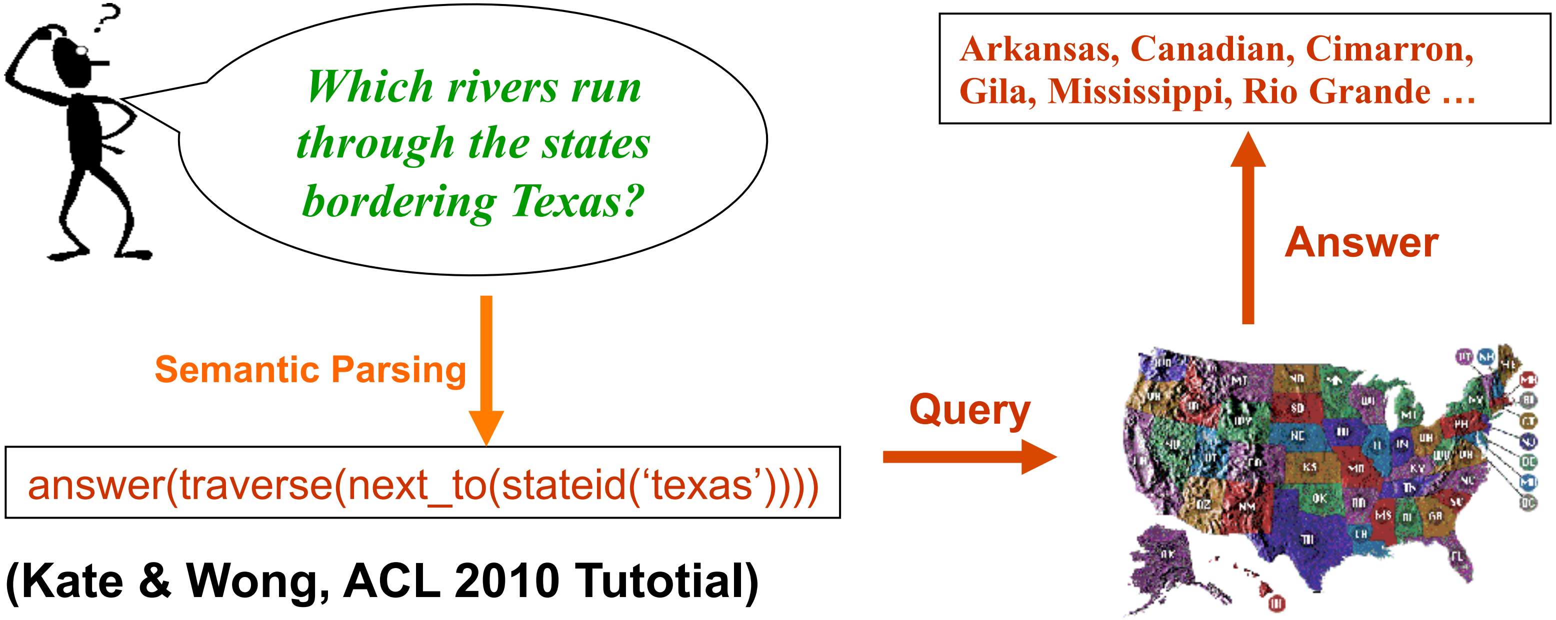
$$\hat{e} = e\left(\operatorname*{argmax}_{D \; s.t. \; s(D) \equiv s} w(D) \right)$$

Outline

- Background
- Rule extraction algorithm
- Modeling
- Experiments
- Conclusion

Experiments

- Dataset: GEOQUERY
  - 880 English questions with corresponding Prolog logical forms



Which rivers run through the states bordering Texas?

Semantic Parsing

answer(traverse(next_to(stateid('texas'))))

Query

Answer

Arkansas, Canadian, Cimarron, Gila, Mississippi, Rio Grande …

(Kate & Wong, ACL 2010 Tutotial)

Experiments

- Dataset: GEOQUERY
  - 880 English questions with corresponding Prolog logical forms
- Evaluation metrics

$$precision = \frac{|C|}{|G|}, recall = \frac{|C|}{|T|},$$

$$F - measure = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

Experiments

System	P	R	F
Independent Test Set			
Z&C 2005	***96.3***	79.3	87.0
Z&C 2007	95.5	83.2	88.9
Kwiatkowksi, *et al.* (2010)	94.1	85.0	89.3
Cross Validation Results			
Kate *et al*. (2005)	89.0	54.1	67.3
Wong and Mooney (2006)	87.2	74.8	80.5
Kate and Mooney (2006)	93.3	71.7	81.1
Lu *et al.* (2008)	89.3	81.5	85.2
Ge and Mooney (2005)	95.5	77.2	85.4
Wong and Mooney (2007)	92.0	86.6	89.2
this work	93.0	***87.6***	***90.2***

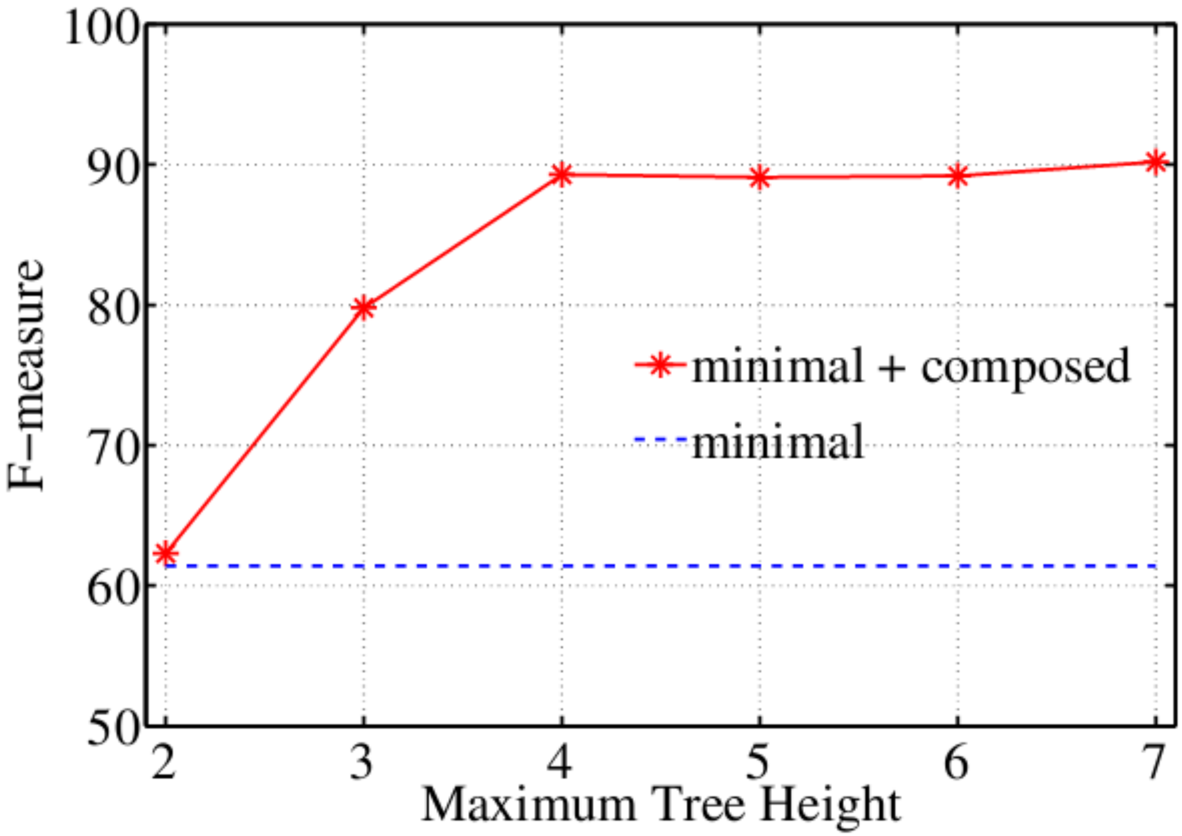
Experiments

- F-measure for different languages

System	**en**	**ge**	**el**	**th**
Wong and Mooney (2006)	77.7	74.9	78.6	75.0
Lu *et al.* (2008)	81.0	68.5	74.6	76.7
Kwiatkowksi, *et al.* (2010)	82.1	***75.0***	73.7	66.4
Jones *et al*. (2005)	79.3	74.6	75.4	***78.2***
this work	***84.2***	74.6	***79.4***	76.7

* en – English, ge – German, el – Greek, th – Thai

Experiments



Advantages

- Feasible to extract rules with varying granularities in a principled way
  - The widely used dataset only has 880 training examples
- Alleviating the data sparseness problem
  - Treating atomic logical form tokens as tree nodes instead of context free grammar (CFG) production
- Robust to the nonisomorphism between NL sentences and logical forms

Outline

- Background
- Rule extraction algorithm
- Modeling
- Experiments
- Conclusion

Conclusion

- We have presented an extended GHKM algorithm for inducing λ-SCFG and achieved state-of-the-art performance
- Future work
  - Better alignment model
  - Investigate tree binarization to further improve rule coverage
  - Use EM or Monte Carlo methods to better estimate λ-SCFG rule probabilities