Representing Meaning with a Combination of Logical and Distributional Models

Computational Linguistics, Volume 42, Issue 4 - December 2016

I. Beltagy, Stephen Roller, Penxiang Chen, Katrin Erk, Raymond J. Mooney

Presented by Michal Porubcin

Introduction: Logical vs Distributional Semantics

Logic-based representations:

+ Encompass negation, quantifiers, entities
+ Standardized inference mechanisms
- Coverage problems from manually constructed dictionaries
- Fail to capture graded aspect of meaning (binary)

Distributional models:

+ Contextual similarity -> semantic similarity
- Do not support logical inference

Introduction: Logical vs Distributional Semantics

“The case for abandoning the categorical view of competence and adopting a probabilistic model is at least as strong in semantics as it is in syntax.” -- van Eijck and Lappin (2012)

“Meaning is about truth... Meaning is also about a community of speakers and how they use language” -- Beltagy et al. (2016)

Task - Recognizing Textual Entailment (RTE)

Text T {entails, contradicts, neutral} Hypothesis H

Not logical entailment; labels provided by human annotators

SICK dataset

- Entailment

T: A man and a woman are walking together through the woods.
H: A man and a woman are walking through a wooded area.

- Contradiction

T: Nobody is playing the guitar
H: A man is playing the guitar

- Neutral

T: A young girl is dancing
H: A young girl is standing on one leg

Hybrid Approach

1) **First-order logic**: primary meaning representation
2) **Distributional information**: weights for logical rules
3) **Markov Logic Networks (MLN)**: inference

$$\forall x.grumpy(x) \rightarrow sad(x) \mid f(sim(\vec{grumpy}, \vec{sad}))$$

$$\forall x.\ ogre(x) \Rightarrow grumpy(x) \mid 1.5$$

$$\forall x, y.\ (friend(x, y) \wedge ogre(x)) \Rightarrow ogre(y) \mid 1.1$$

Example (3) -- Markov Logic Networks

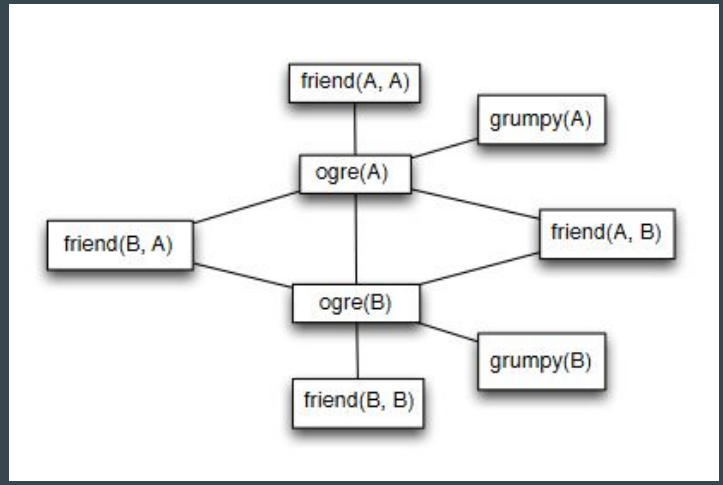
Markov networks: undirected graphical models

MLNs construct Markov Networks based on weighted FOL formulas

Example: Two constants: Anna (A) and Bob (B)

$$\forall x.\, ogre(x) \Rightarrow grumpy(x) \mid 1.5$$

$$\forall x, y.\, (friend(x, y) \wedge ogre(x)) \Rightarrow ogre(y) \mid 1.1$$



High-Level Architecture + Task Representation

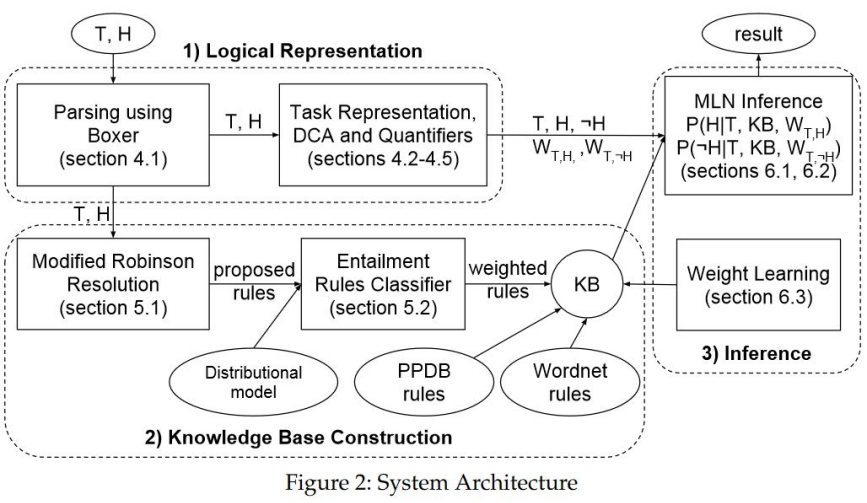


Figure 2: System Architecture

[i] $P(H|T, KB, W_{T,H})$

[ii] $P(\neg H|T, KB, W_{T,\neg H})$

[i] high, [ii] low: **Entailment**
[i] low, [ii] high: **Contradiction**
[i] similar to [ii]: **Neutral**

Parsing with Boxer

Rule-based semantic analysis system

Input: CCG parse

flights	to	boston
N	$(N\backslash N)/NP$	NP
$\lambda x.flight(x)$	$\lambda y.\lambda f.\lambda x.f(x) \wedge to(x,y)$	*boston*
	$(N\backslash N)$ (>)	
	$\lambda f.\lambda x.f(x) \wedge to(x, boston)$	
N (<)		
$\lambda x.flight(x) \wedge to(x, boston)$		

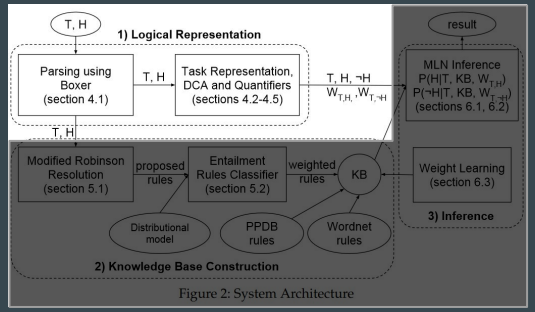


Figure 2: System Architecture

More information:

- Boxer: https://www.aclweb.org/anthology/W15-1841.pdf
- CCGs: http://www.cs.tau.ac.il/~joberant/teaching/Talks/dor.pdf
- NDF: http://www.coli.uni-saarland.de/courses/incsem-12/neodavidsonian.pdf

Parsing with Boxer

Neo-Davidsonian framework (NDF)

Example: {An ogre loves a princess}$_{NL}$

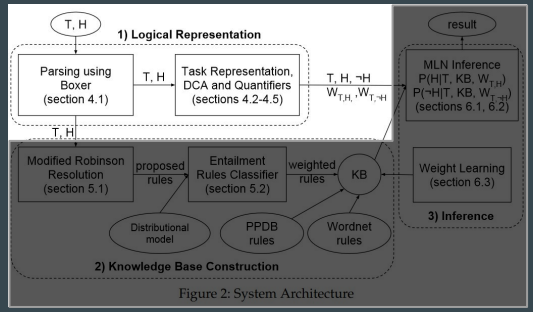


Figure 2: System Architecture

Note: this is not a rule! (no implication)

$$\exists x, y, z.\ ogre(x) \wedge agent(y, x) \wedge love(y) \wedge patient(y, z) \wedge princess(z)$$

Domain Closure Assumption (DCA)

1 to 1 relationship between objects in domain and named constants of F, i.e.

There are no objects in the universe other than the named constants.

MLNs only handle finite set of constants

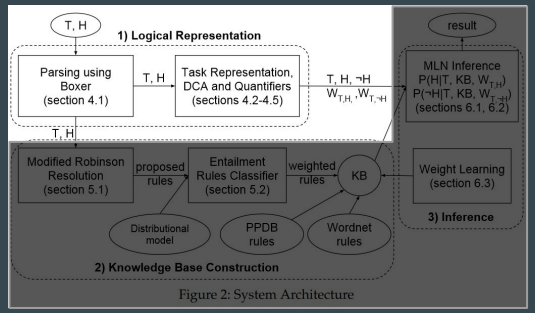
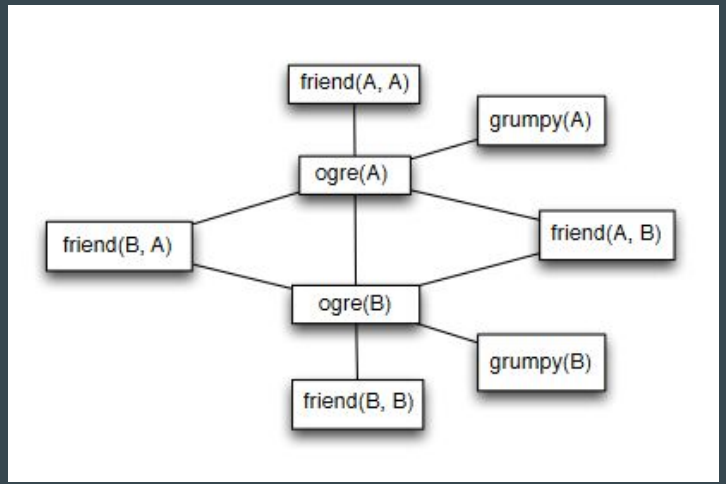


Figure 2: System Architecture



Closed World Assumption (CWA)

Everything is false unless stated otherwise

Assign ground atoms low prior probabilities

Benefits:

- Entailment not a result of world knowledge
- Inference less sensitive to domain size
- Computational efficiency

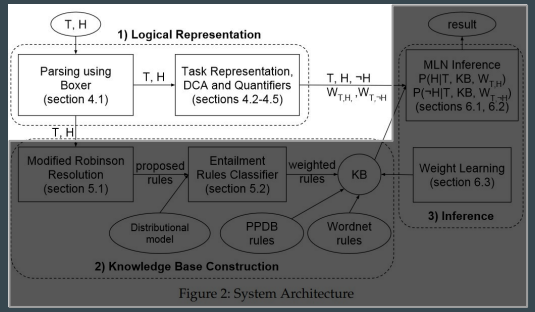


Figure 2: System Architecture

$$P(H|T, KB, W_{T,H})$$

$$P(H|KB, W_{T,H})$$

Multiple Parses

More robust to downstream errors

Generate two parses for both T and H -> T_1, T_2, H_1, H_2

Compute probabilities for all combinations of H given T:

T_1, T_2, H_1, H_2 (and $\neg H_1$, $\neg H_2$)

Thresholding classifier (in stage 3) trained to take in all resulting probabilities as features

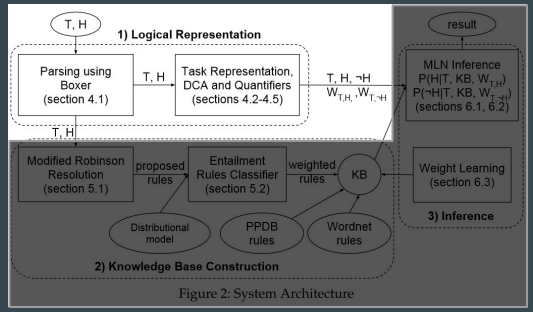


Figure 2: System Architecture

Knowledge Base Construction

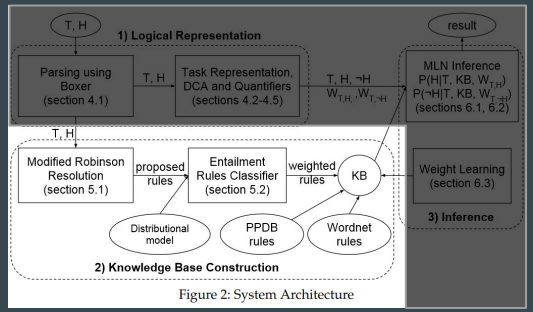


Figure 2: System Architecture

What we have: modified meaning repr. from Boxer

$$\exists x, y, z.\ ogre(x) \wedge agent(y, x) \wedge love(y) \wedge patient(y, z) \wedge princess(z)$$

What we want: weighted rules

$$\forall x, y.\ (friend(x, y) \wedge ogre(x)) \Rightarrow ogre(y) \mid 1.1$$

3 Groups of rules:

1. Classified rules from MRR
2. Wordnet
3. PPDB

Rules as Training Data

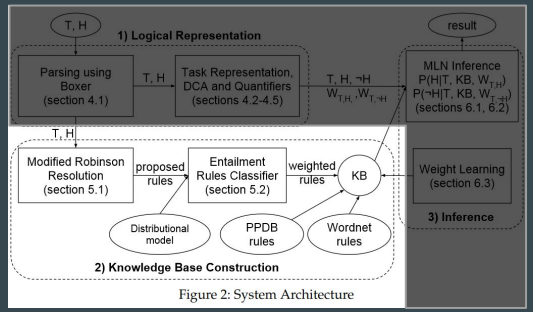


Figure 2: System Architecture

Convert MRR rules into textual rules (replace Boxer predicate with NL word)

Use {entailment, contradiction, neutral} labels on RTE task to derive labels for individual rules

[i] Entailment: All rules labeled entailment
[ii] Neutral: Compare against rules from [i] + manual annotation
[iii] Contradiction: Assume either T or H is negated

$$T \wedge r_1 \wedge \ldots \wedge r_n \Rightarrow H$$

Lexical Entailment Rule Classifier

Predict entailment between single words

Supervised classification with below features:

- Wordform: same lemma, same POS, etc
- WordNet: synonymy, antonymy, hypernymy, etc
- Distributional: histogram binning of cosines
- Asymmetric: use dependency space generated with distributional features

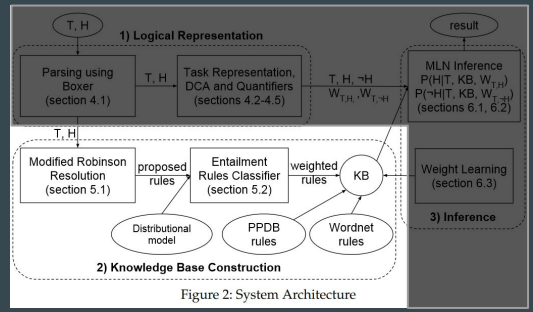


Figure 2: System Architecture

$$\forall x.ogre(x) \Rightarrow monster(x)$$

The idea is to generate features from the relationship between LHS and RHS

More Info On Distributional Features

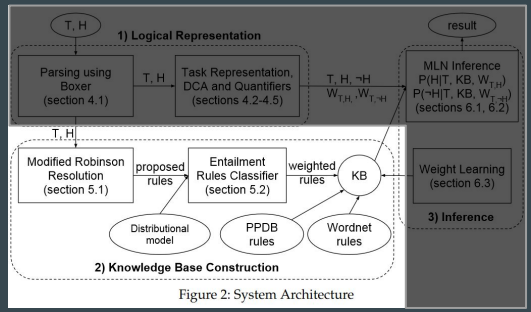


Figure 2: System Architecture

Preprocessing:

- BNC, ukWaC, and Wikipedia fed into Stanford CoreNLP
- Keep only content words {NN, VB, RB, JJ} appearing at least 1000 times

Bag of Words vectors:

- Skip-Gram Negative Sampling (window size: 20)

Dependency Vectors:

- Extract tuples from Stanford Collapsed CC Dependency graphs
- Build vector space with (lemma/POS) as rows and (relation, context/POS) as columns

Cosine similarity in either space as features

Extension to Phrases

Many of rules from MRR have multiple words (phrases)
e.g. little boy -> child

1. Compositional model
2. Greedy word aligner
   - Find pairs of words from LHS/RHS most similar in distributional space
   - Compute base features based on results of alignment

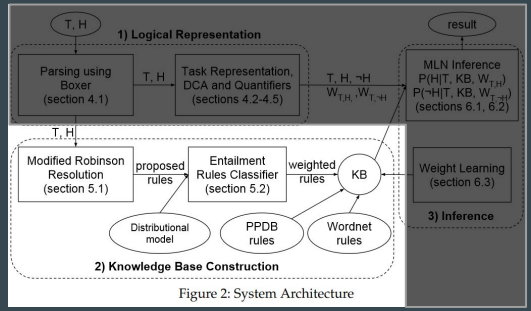


Figure 2: System Architecture

More Info On Compositional Model

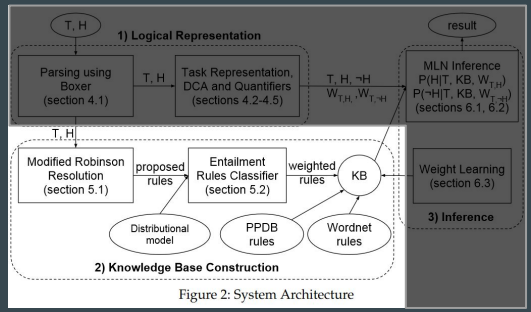


Figure 2: System Architecture

Preprocessing:

- BNC, ukWaC, and Wikipedia fed into Stanford CoreNLP
- Keep certain dependencies {amod, nsubj, dobj, pobj, acomp} and combine governor and dependent words into phrases
- Governor and dependent among 50K most frequent words in corpus'
- Word representation:
  - Vector: contexts
  - Several Matrices: for each dependent type

$$\vec{chase} + \overset{\square_s}{chase} \times \vec{cat} + (\vec{chase} + \overset{\square_o}{chase} \times \vec{dog})$$

Rule Group 2: WordNet

Substring matching with T+H pair and WordNet to find relevant rules

Represent as logical rules

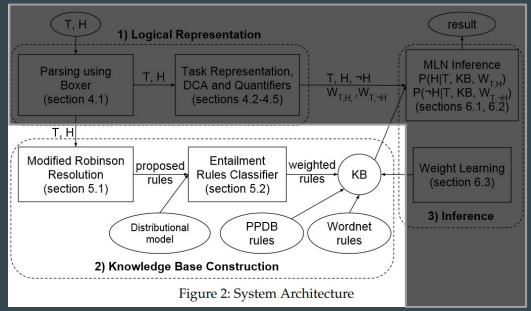


Figure 2: System Architecture

Synonymy: $\forall x.\, man(x) \Leftrightarrow guy(x)$
Hypernymy: $\forall x.\, car(x) \Rightarrow vehicle(x)$
Antonymy: $\forall x.\, tall(x) \Leftrightarrow \neg short(x)$

Rule Group 2: PPDB

Substring matching with T+H pair and PPDB to find relevant rules

Rule-based translation of PPDB rules to logical rules:

1. (Assume conjunction of positive atoms in PPDB)
2. Break down PPDB rule into predicates
3. Add Boxer meta-predicates based on Boxer parse of T+H pair

Rule-based binding of variables in LHS to RHS:

1. Manually define paraphrase rule templates for PPDB
2. Existentially quantify unbound RHS variables

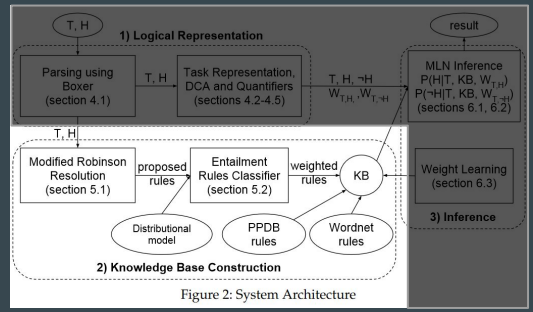


Figure 2: System Architecture

Rule Group 3: Handcoded Rules

Handful of manually added rules

For SICK dataset, lexical rules where one side is “nobody”

$$nobody \Leftrightarrow \neg\, somebody$$

$$nobody \Leftrightarrow \neg\, person$$

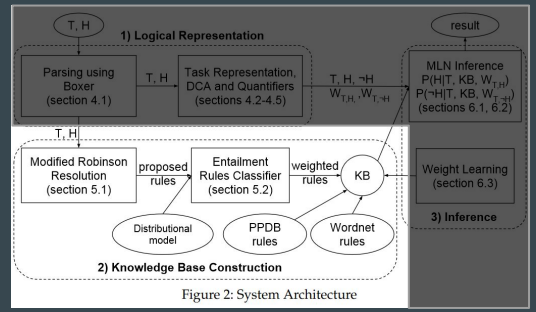


Figure 2: System Architecture

Weight “Learning”

Weights from different sources may be on different scales

Grid search to find appropriate scale factors

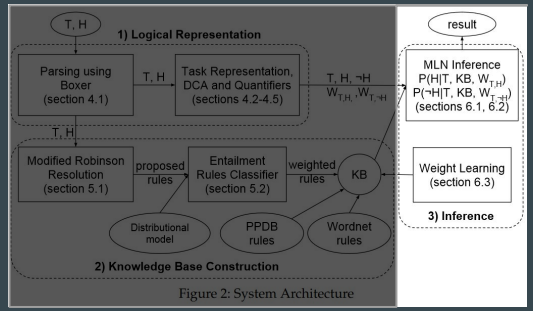


Figure 2: System Architecture

$$MLNweight = scalingFactor \times ruleWeight$$

$$MLNweight = scalingFactor \times \log(\frac{ruleWeight}{1 - ruleWeight})$$

Final Entailment Rules Classifier

10 fold cross validation on annotated training set

Logistic regression with L2 regularization

Other models tried: Decision Trees, SVMs (various kernels)

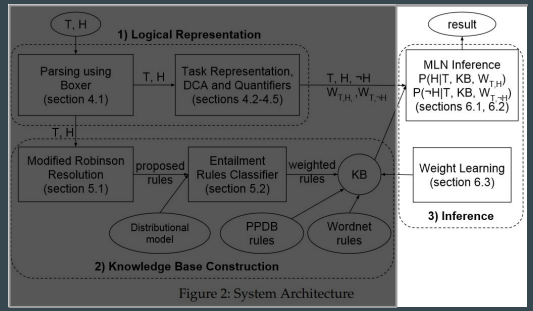


Figure 2: System Architecture

MLN Construction

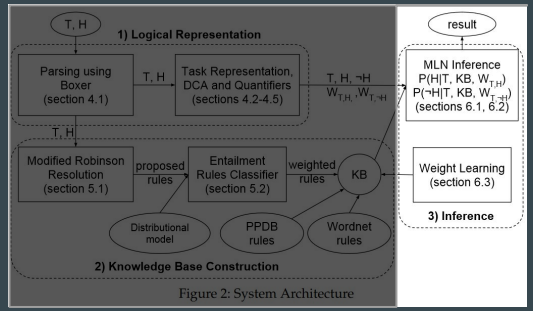


Figure 2: System Architecture

What we have:

1. Task representation
2. Weighted rules

$$L_{A,B} = \{ogre(A), ogre(B), grumpy(A), grumpy(B), friend(A, A), friend(A, B), friend(B, A), friend(B, B)\}$$

What we want:

1. Markov Logic Network
2. Inference: an entailment prediction

$$\forall x.\, ogre(x) \Rightarrow grumpy(x) \mid 1.5$$

$$\forall x, y.\, (friend(x, y) \wedge ogre(x)) \Rightarrow ogre(y) \mid 1.1$$

More information:

- MLNs: https://homes.cs.washington.edu/~pedrod/papers/mlj05.pdf

MLN Construction

Given [i] constants Anna (A) and Bob (B) and [ii] input formulas

1. Generate all ground atoms (nodes in graph)
2. Connect two nodes if co-occur in grounding of input formula

Note: Each clique corresponds to grounding of a rule.

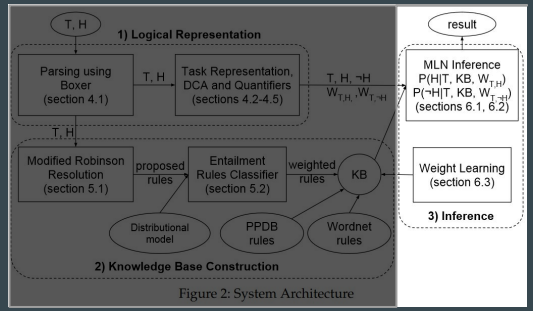
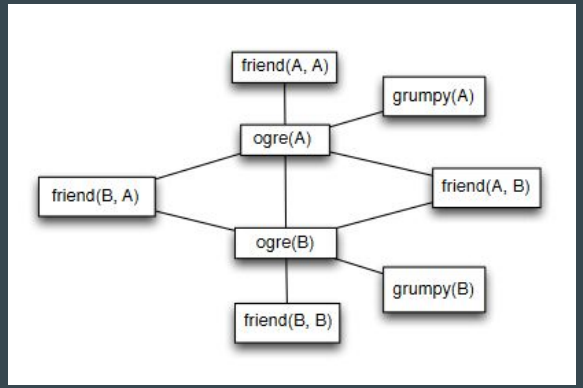


Figure 2: System Architecture

$$\forall x.\, ogre(x) \Rightarrow grumpy(x) \mid 1.5$$

$$\forall x, y.\, (friend(x, y) \wedge ogre(x)) \Rightarrow ogre(y) \mid 1.1$$

$$L_{A,B} = \{ogre(A), ogre(B), grumpy(A), grumpy(B), friend(A, A), friend(A, B), friend(B, A), friend(B, B)\}$$



MLN Inference

Variable assignment assigns {T,F} to each node -> a “world”

Variable assignment makes underlying ground rules true or false

Clique potential: function that assigns a value to each clique

Compute probability of a world

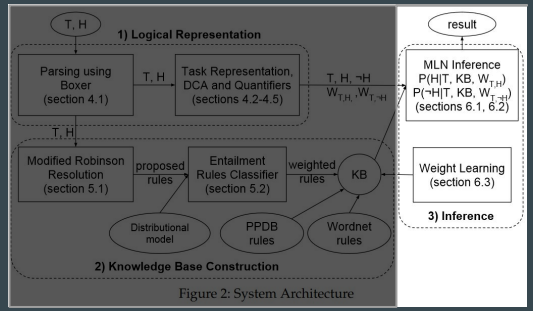


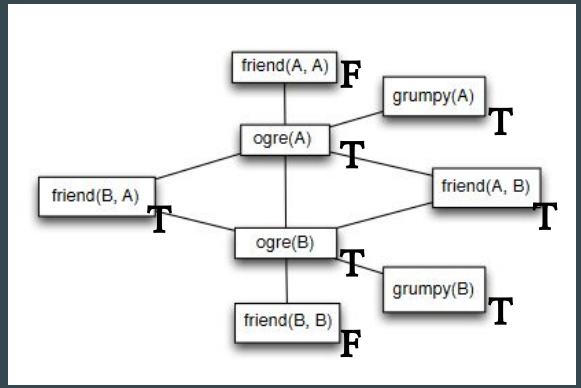
Figure 2: System Architecture

$$\forall x.\, ogre(x) \Rightarrow grumpy(x) \mid 1.5$$

$$\forall x, y.\, (friend(x, y) \wedge ogre(x)) \Rightarrow ogre(y) \mid 1.1$$

$$L_{A,B} = \{ogre(A), ogre(B), grumpy(A), grumpy(B), friend(A, A), friend(A, B), friend(B, A), friend(B, B)\}$$

$$P(X = x) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(x)\right)$$



Inference on Complex Formulas

Probability of a formula: Sum of probabilities of possible worlds that satisfy it

Problem: Current MLN implementations only support probabilities of ground atoms

Naive fix: Add complex formula to MLN with brand new ground atom with infinite weight

Problem: Backwards implication intractable

Better idea: Compute partition Z with and without H

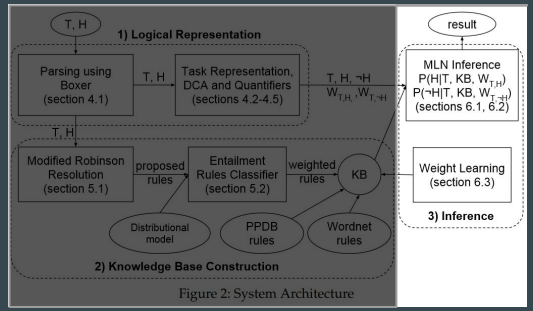


Figure 2: System Architecture

$$\sum_X \quad P(X = x) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(x)\right)$$

$$H \Leftrightarrow result(D) \mid \infty$$

$$P(H \mid KB) = \frac{Z(KB \cup \{(H, \infty)\})}{Z(KB)}$$

How CWA optimizes queries

H (query) equivalent to disjunction of all possible queries

Any ground atoms NOT inferred from T or T $\wedge$ KB are false

In practice, eliminates O(c^v) behavior from # of ground clauses of H

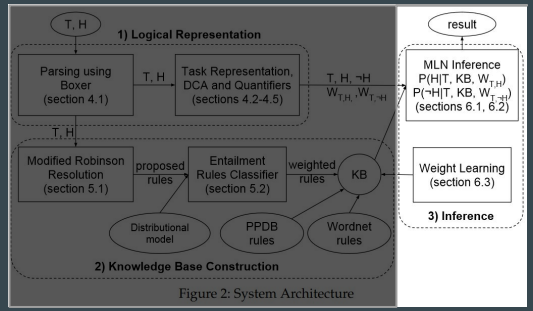


Figure 2: System Architecture

$$T : ogre(O) \wedge agent(S, O) \wedge snore(S)$$
$$KB : \forall x.\, ogre(x) \Rightarrow monster(x)$$
$$H : \exists x, y.\, monster(x) \wedge agent(y, x) \wedge snore(y)$$

Final MLN Classifier

Learn thresholds for probability of $\neg H_1$ and $\neg H_2$

SVM classifier with LibSVM's default parameters.

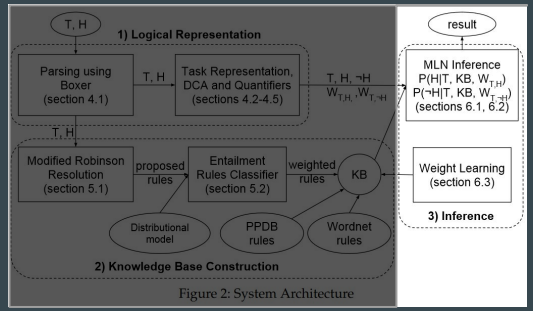


Figure 2: System Architecture

$$P(H|T, KB, W_{T,H})$$

$$P(\neg H|T, KB, W_{T,\neg H})$$

Evaluation: Entailment Rules Classifier

Feature set	Intrinsic	RTE Train	RTE Test
Always guess neutral	64.3	73.9	73.3
Gold standard annotations	100.0	95.0	95.5
Base only	64.3	73.8	73.4
Wordform only	67.3	77.0	76.7
WordNet only	75.1	81.9	81.3
Dist (Lexical) only	71.5	78.7	77.7
Dist (Phrasal) only	66.9	75.9	75.1
Asym only	70.1	77.3	77.2
All features	79.9	84.0	83.0

Table 3: Cross-validation accuracy on Entailment on all rules

Feature set	Intrinsic	RTE Train	RTE Test
Always guess neutral	56.6	69.4	69.3
Gold standard annotations	100.0	93.2	94.6
Wordform only	57.4	70.4	70.9
WordNet only	79.1	83.1	84.2
Dist (Lexical) only	68.8	76.3	76.7
Asym only	76.8	78.3	79.2
All features	84.6	82.7	83.8

Table 4: Cross-validation accuracy on Entailment on lexical rules only

Feature set	Intrinsic	RTE Train	RTE Test
Always guess neutral	67.8	72.5	72.7
Gold standard annotations	100.0	91.9	92.8
Base only	68.3	73.3	73.6
Wordform only	72.5	77.1	77.1
WordNet only	73.9	78.3	77.7
Dist (Lexical) only	72.9	77.0	76.5
Dist (Phrasal) only	71.9	75.7	75.3
All features	77.8	79.7	78.8

Table 5: Cross-validation accuracy on Entailment on phrasal rules only

Evaluation: Entire System

Components Enabled	Train Acc.	Test Acc.
logic + cwa + coref	73.8	73.4
logic + cwa + coref + eclassif	84.0	83.0
+ handcoded	84.6	83.2
+ handcoded + multiparse	85.0	83.9
+ handcoded + multiparse + hyp	85.6	83.9
+ handcoded + multiparse + hyp + wlearn	85.7	84.1
+ handcoded + multiparse + hyp + wlearn_log	85.9	84.3
+ handcoded + multiparse + hyp + wlearn_log + mem	93.4	**85.1**
+ handcoded + multiparse + hyp + wlearn_log + mem + ppdb	93.4	84.9
current state of the art (Lai and Hockenmaier 2014)	–	84.6

Table 8: Ablation experiment for the system components with **eclassif**, and the best performing configuration

Limitations

- Low usage of distributional models
- Rule-based approach for determining coreference
- Weights for inference rules not dependent on context
- Robinson Resolution does not handle duplicate words
- No general algorithm for when to extend a rule
- Manual annotation of training data for Entailment Rule Classifier
  - Assumption about contradiction pairs in SICK dataset specifically
- Rule-based PPDB features
- Extra hand coded rules

Thank you!