What did we learn over the course?

Putting All Together within Applications

Dialogue System: Curt

Home Automation

NLP System

Textual Entailment Recognition

---

1 Slides are partially due to J. Bos and P. Blackburn course on "An Introduction to Computational Semantics"
Overview

1. What did we learn over the course?
2. Putting All Together within Applications
3. Dialogue System: Curt
4. Home Automation NLP System
5. Textual Entailment Recognition
1. What did we learn over the course?

2. Putting All Together within Applications

3. Dialogue System: Curt

4. Home Automation NLP System

5. Textual Entailment Recognition
What did we learn so far?

What do we do in Computational Semantics:

- Construction of a **Semantic Representation**
- Semantic Resolution, Interpretation
- Semantic Inference
Sample Computation Semantic Workflow

- Sentence $\rightarrow$ Syntactic Parse Tree $\rightarrow$ Logical Form $\rightarrow$ Interpretation
- S: Bill likes Simpsons

\[
\text{like}(\text{Bill}, \text{Simpsons})
\]

- NP: Bill
- VP: likes Simpsons
- V: likes
- NP: Simpsons

- Lexicon provides leaf informations (Lexical Semantic)
- Syntactic tree provides ?-substitution
- FOL-inference tools provide interpretation
Sample Interpretation of an Utterance

Bill likes a cartoon (2)

- FOL: $\exists x (\text{cartoon}(x) \land \text{like}(Bill, x))$
- Elements of a domain:
  - *Bill, Simpsons*
- Concepts of a domain:
  - *like, cartoon*
  - Domain Knowledge: *Bill may not be a patient of like, Simpsons is a cartoon, Bill is not a cartoon*
We now have some answers to the two fundamental questions with which this course began:

- We know how to build DRS-representations for natural language with the help of the lambda calculus, where DRS-representation also takes care of some scope ambiguities. We know how to build first-order representation for DRSs.
- We have learned about the process of performing inference with first-order representations. In particular, we have learned something about theorem provers, model builder and model checking:
  - querying, informativity checking, and consistency checking.
- Now it’s time to have a little fun, bring the various bits and pieces together, and see what they can do for us.
1. What did we learn over the course?

2. Putting All Together within Applications

3. Dialogue System: Curt

4. Home Automation NLP System

5. Textual Entailment Recognition
Sample Applications

- Dialogue System – Curt
- Home Automation NLP System
- Textual Entailment Recognition

Gliederung

1. What did we learn over the course?
2. Putting All Together within Applications
3. Dialogue System: Curt
4. Home Automation NLP System
5. Textual Entailment Recognition
Chatting with CURT

- CURT-dialogue system is programmed in prolog
- It contains modules that implement
  - lexical, semantical database
  - lambda calculus,
  - drs-building,
  - model checking, theorem proving and model building programs
  - interface that allows simple interaction with the user.
CURT’s Functionality

- The user can extend CURT’s knowledge by entering English sentences and then query CURT about its acquired knowledge.
- Different versions of CURT are possible, that represent the various stages of development in CURT’s life.
The backbone of the CURT system is:

- a minimum of inference tools included (namely none).
- The user can enter English sentence and check the knowledge state of CURT
But Baby Curt is having some problems!

- As Baby Curt doesn’t impose on any inference tools, it accepts discourses like:
  
  USER: "Mia smokes and does not smoke."
  CURT: "OK"
  USER: "Vincent is a man."
  CURT: "OK"
  USER: "Mia likes every man."
  CURT: "OK"
  USER: "Mia does not like Vincent."
  CURT: "OK"

- So we need to do something here to make CURT deal with inconsistent information.
Clever CURT only accepts consistent English sentences.
The user is also able to view the models generated for the English sentences given to CURT.
Ambiguous input might eliminate some of the inconsistent interpretations.
Clever Curt uses the model builder MACE to check consistency, and the theorem prover OTTER to check inconsistency of interpretations.
But Clever Curt has its problems too!

- Although Clever Curt rejects inconsistent English sentences, it has no clue how to distinguish old from new information:

  USER: "Mia smokes."
  CURT: "OK"
  USER: "Mia smokes."
  CURT: "OK"
  USER: "Vincent knows every boxer."
  CURT: "OK"
  USER: "Butch is a boxer."
  CURT: "OK"
  USER: "Vincent knows Butch."
  CURT: "OK"

- So we extend CURT to deal with uninformative information.
CURT III: Sensitive Curt

- Sensitive Curt is clever! (it knows what consistency means)
- Sensitive Curt will also note uninformative contributions of the user.
  - It does so by using MACE and OTTER to check whether new contributions are informative with respect to the readings CURT has in its memory.
More Improvement Possibilities

- Quantifying noun phrases might induce different interpretations
- Some of these readings can be logically equivalent, but neither clever nor sensitive CURT notice:
  USER: A boxer kills a man
  CURT: OK.
  USER: readings
  1. $\exists a(\text{boxer}(a) \land \exists b(\text{man}(b) \land \text{kill}(a, b)))$
  2. $\exists a(\text{man}(a) \land \exists b(\text{boxer}(b) \land \text{kill}(b, a)))$
- Using theorem proving techniques once again, we make CURT filter out logical equivalent interpretations
CURT IV: Laconic Curt

- Laconic Curt eliminates equivalent readings using the Theorem Prover OTTER.
- Given $R1$ and $R2$ from a set of readings, $R1$ and $R2$ will be replaced by $R1$ if OTTER finds a proof for $((R1 \rightarrow R2) \land (R2 \rightarrow R1))$
- Note that this is not necessarily efficient.
Laconic’s problems

Still, Curt is quite dumb!

- The CURTs developed so far possess no world knowledge at all!
- For instance, Sensitive Curt does not know that Mia is a woman, nor does it know that men and women denote different kinds of entities.

USER: Mia is a woman.
CURT: OK.
USER: Mia is a man.
CURT: OK.

- So we should supply additional information about the logical symbols we use.
- How?
Adding Lexical Knowledge:

- Let’s focus on nouns and adjectives
  - Hypernyms (example: vehicle is a hypernym of car)
    - In first-order logic: $\forall x (\text{car}(x) \rightarrow \text{vehicle}(x))$
  - Antonyms (example: males are disjoint from females)
    - In first-order logic: $\forall x (\text{male}(x) \land \neg \text{female}(x))$
- Put this information in the lexicon
  - the required background knowledge is computed from the lexicon on the basis of CURT’s input (so not all available knowledge is used all of the time)
Hyponym Database (MiniWordNet)

- thing
  - event
  - entity
    - object
    - organism
Knowledgeable problems

Now Curt is knowledgeable, but not able to share its knowledge...

- We not only want to assert information to CURT’s database.
- We also want to query information!
- For instance by posing ”Who likes Mia” CURT should return all named entities that like Mia (according to its current model).
- Model Checker for first-order models can be used to accomplish this task.
CURT VI: Super Curt

- Questions introduce lambda-expressions
- Model Checkers can be used to extract information from the model
- For instance "Who walks" gives us lambda(X,walk(X))
- Then we try to evaluate this expression in the current model wrt assignment g(X,Answer)
- If we are successful Answer will unify with an element of the domain
- Having a member of the domain at our disposal, we realize CURT’s answer as a:
  - proper name if it is named in the model
  - indefinite noun phrase if there is a one-place noun relation
Possibilities for Improving Super Curt

- Super Curt combines several inference tools (Theorem Prover, Model Checker, Model Builder) for making responses
- The current architecture implements a purely sequential approach
- Especially for consistency/informativity checking, we might want to run theorem provers and model builders in parallel:
  - While the theorem prover tries to prove $\neg \phi$, the model builder attempts to construct a model for $\phi$
- One might also consider spreading the computational power by running computationally demanding tasks on different machines
And let the fun begin:

- Demo
What did we learn over the course?

Putting All Together within Applications

Dialogue System: Curt

Home Automation NLP System

Textual Entailment Recognition
Home Automation Dialogue System

- User
- Dialogue System
  - Speech Recognition \(\Rightarrow\) *Logic Form Representation* \(\Rightarrow\) Inferencing Component \(\Rightarrow\) Generation and Synthesis of Speech or Actions

*Logic Form Representation* is U/DRS
- Inferencing Component
  - translates DRS into FOL
  - invokes Model Checkers, Model Builders, and Theorem Provers
  - interprets inference tools output into actions or speech acts

- Does not it look like CURT?

---

3"An Inference-based approach to dialogue system design" by J. Bos and T. Oka
"Smart" room consists of:

- three lights (red, blue, black)
- radio

Possible actions:

- turn on/off
- switch on/off
To make it work

Domain axioms about actions and state changes must be specified:

- "turning a device on" and "switching a device on" are kinds of ⇒ "power-on" actions
  - where a power-on action requires a device to be off in the current state and causes them to be on in the resulting state
- "frame-problem" related axioms
Possible commands

- turn the red light on
- turn it off
- switch on all devices
- turn off all devices except the radio
- turn on the red or blue light
- turn off another light...
System in work

Possible system reaction is

- response
  - expresses understanding (for consistent instructions), or
  - signals contradiction (for inconsistent instructions)

- performance of requested action/s
What did we learn over the course?

Putting All Together within Applications

Dialogue System: Curt

Home Automation NLP System

Textual Entailment Recognition
RTE problem consists of:

- verifying whether some text $T$ entails a hypothesis $H$

---

4"Recognising Textual Entailment with logical inference" by J. Bos and K. Markert
**True Example**

**T:** In 1998, the General Assembly of the Nippon Sei Ko Kai (Anglican Church in Japan) voted to accept female priests.

**H:** The Anglican church in Japan approved the ordination of women.
False Example

**T:** The city Tenochtitlan grew rapidly and was the center of the Aztec’s great empire.  
**H:** Tenochtitlan quickly spread over the island, marshes, and swamps.
What is available to us?:
- building DRSs for T and H
- translation of DRSs into FOL

Now note that
- T implies H (shows entailment)
- T+H are inconsistent (shows no entailment)

Remember that we can use theorem provers and models builders for consistency/inconsistency verification.
Theorem prover finds a proof for:

* \( \text{FOL(DRS}(T)) \rightarrow \text{FOL(DRS}(H)) \Rightarrow \) (logical) entailment

* \( \neg \text{FOL(DRS}(T);\text{DRS}(H)) \Rightarrow \) T definitely does not entail H (assuming T and H are themselves consistent)
Model builder finds a model for:

⋆ \( \neg (\text{FOL}(\text{DRS}(T)) \rightarrow \text{FOL}(\text{DRS}(H))) \) says there cannot be a proof for its negation \( \Rightarrow \) no entailment

⋆ \( \text{FOL}(\text{DRS}(T);\text{DRS}(H)) \) shows \( T \) and \( H \) are consistent \( \Rightarrow \) may be entailment
What’s missing in real world?

Background knowledge!

- generic knowledge (possessives, active-passive alternation, spatial knowledge... – manual generation, 20)
- lexical knowledge (MiniWordnet – automatic generation)
- geographic knowledge (automatic generation)
Sum up

- Shallow approaches dominate
- Inference approach is nevertheless at least as good and has a potential for improvement!
Course Sum Up

- Questions?
- Feedback
- Wishes for what’s next?