Practical Verification Strategies for Dialogue Management

Míra Janíček

November 20, 2008

Verification × Grounding

- in general, **grounding** is the process of converging to common knowledge
- in practical dialogue systems, however, it is often reduced to **verification** (confirmation) of system's recognition of user utterances

Schematically

- two dialogue participants, S and H
- S has just uttered or is uttering *utt* to H:

$$S \xrightarrow{utt} H$$

- H gives S **feedback** *fb*:

$$S \xleftarrow{fb} H$$

fb in Human-Human Communication

- **level of action**
  - **contact** – S and H have established a channel of communication
  - **perception** – H perceives *utt*
  - **understanding** – H has understood *utt*
  - **acceptance** – H has accepted and integrated *utt*
- **polarity**
  - **positive** – positive signal that the given level is achieved
  - **negative** – negative signal (e.g. no understanding)
  - **checking** – between the two: H has a hypothesis, but needs to check if it is valid

fb in Human-Computer Communication

- in order to verify that the system has understood *utt* correctly, it needs to convey its understanding of *utt* back to the user and ask him if it is OK
- we want the dialogue to be as fluent as possible
- if there is a problem, whose fault is it?

Either of the parties may be responsible:

- the system: poor speech recognition performance
- the user: incorrect assumptions about the system's capabilities

Speech Recognition Errors

- good thing: the speech recognizer spits out the best hypothesis **and its score** (whatever it may be)

$\Rightarrow$ the system is aware of recognition quality (the errors)

- even better: the system may use the score for generating *fb*
  - idea: the more we believe what we've heard, the less confirmation we require, and the more fluent dialogue we achieve

Confirmation as a *fb*

- use confirmation as a *fb*
  +
  be as fluent as possible

- expliciteness of the confirmation $\approx$ low score of perceived *utt*

Types of Confirmation

(San-Segundo et al., 2001)

- **no confirmation** ... for yes/no answers
- **implicit confirmation**
  - = the understood value of *utt*
  - + prompt for next item
- **semi-implicit confirmation**
  - = the understood value of *utt*
  - + command for correction
  - + prompt for next item
- **explicit confirmation**
  - = the understood value of *utt*
  - + explicit request for confirmation
- **item value rejection**

Determining Confidence

- (San-Segundo et al., 2001) define 4 confidence levels (CF):

  w ... # of correctly recognized words in *utt*
  e ... # of errors

- $\mathrm{CL}(utt) = 1 \quad \Leftarrow \quad w \gg e$ ... very high confidence
- $\mathrm{CL}(utt) = 2 \quad \Leftarrow \quad w > e$ ... high confidence
- $\mathrm{CL}(utt) = 3 \quad \Leftarrow \quad w \approx e$ ... low confidence
- $\mathrm{CL}(utt) = 4 \quad \Leftarrow \quad w \ll e$ ... very low confidence

Example: Implicit Confirmation

- example:

  System: Which month do you want to leave?
  User: July.
  System: You want to leave in July.
  Which day do you want to leave?

- if the follow-up question is semantically related to the confirmation, we may combine the two:

  System: Which month do you want to leave?
  User: July.
  System: Which **day of July** do you want to leave?

- this produces a more fluent dialogue.

Example: Semi-implicit Confirmation

- example:

System:	Which month do you want to leave?
User:	July.
System:	You want to leave in July.
	In case of error, please correct. ⟵
	Which day do you want to leave?

- lengthy and unfriendly.

Example: Explicit Confirmation

- example:

  System: Which month do you want to leave?
  User: July.
  System: Do you want to leave in July?

- we may split the sentence into two:

  System: Which month do you want to leave?
  User: July.
  System: You want to leave in July.
  Is it correct?

- a yes/no answer is expected.

Discussion

- to make the system user-friendly, confirmations should stick to the formulations used by the *User*:

  System: Which day do you want to leave?
  User: This Monday.
  System: Do you want to leave **this Monday**?

- both explicit and (semi-)implicit confirmations contain verbatim reformulation of the *User*'s utterance

  $\Rightarrow$ in a sense, they are all explicit

User's Familiarity with the System

- non-skilled users might not be aware of the system's limitations
  - either its of **language competence** ... i.e. what it understands
  - or its **conceptual** (functional) limits ... i.e. what can it do
- the user may even confuse one for the other
- it is difficult to recognize such states

$\Rightarrow$ we need to give the user some guidelines/help

- but on the other hand we don't want to annoy skilled users

Possible Approach: User Modelling

- the user is given an expertise rating, a **skill level**
- the more skilled user, the less help he gets and the more fluent the dialogue
- at the beginning, the user is assumed to be slightly experienced
- as the dialogue goes on, the user may either
  - "gain skill" if there are no problems
  - "lose skill" if there are

User Modelling: Example

- (San-Segundo et al., 2001) use 4 skill levels, level 1 being the non-experienced and level 4 expert user.
- example:

[the system is in level 3]
System: Say the period of the day you want to travel in.
User: After lunch.

[the system recognizes "in the evening"]
System: Have you said in the evening?
User: No.

[the system decreases the level from 3 to 2]
System: Say the period of the day you want to travel in;
in the morning, in the afternoon or in the evening?
User: In the afternoon.

Possible Approach: Targeted Help

(Hockey et al., 2003)

- a fall-back module that tries to help in cases when the main dialogue system couldn't understand
- two speech recognizers
  - the primary is grammar-based
  - the secondary uses a statistical language model
- Targeted Help gets activated when the primary recognizer rejects and the secondary gets a hypothesis that cannot be parsed

$\Rightarrow$ handles unparsable input

- its output is a help or advice to the user (assumes that the user lacks the knowledge of the system)

Targeted Help: Error Handling

Types of errors handled:

- **endpointing errors**
  - the user pressed the push-to-talk button too late, something is missing at the beginning of the utterance
  - determined by looking at the grammar – is the first word of the utterance a valid initial word in the grammar?
  - if not, an endpointing error is assumed and reported
- **unknown vocabulary**
  - reports "the system doesn't understand the word X"
- **subcategorization mistake**
  - e.g. "zoom in on a car" when it is only possible to "zoom in" and "look at a car" separately
  - **in fact a conceptual error**
  - tries to find the closest valid action, compare it with the proposed one and report it to the user