Assignment 8

L445/L515
Machine Translation
Dialogue Systems

Due Wednesday, December 7

1. Do question #7 from chapter 7 of the draft textbook (p. 272).

2. Do question #6 from chapter 7 of the draft textbook (p. 270–272).

3. There is a corpus of academic transcripts at http://quod.lib.umich.edu/m/micase/ – the Michigan Corpus of Academic Spoken English (MiCASE). Pick a transcript of your choice and answer the following questions:

   (a) What do you notice about the turn-taking (length of time, interruptions, etc.), and what does this tell us about the relationship between the speakers?

   (b) How do you see the speakers establishing common ground? Be specific.

   (c) Do the following for the first four exchanges (i.e., about 8 turns):

   • Divide the speech into utterances
   • Label each utterance with at least one DAMSL tag (http://www.cs.rochester.edu/research/speech/damsi/RevisedManual/)

   If you have a particular hypothesis about the effect of, e.g., the discipline or type of speech event, compare your transcript with some other ones to make your case stronger.

4. Do question #5 from chapter 6 of the draft textbook (p. 224), except you do not actually have to implement your system; simply redesign the templates.

5. Do question #6a from chapter 6 of the draft textbook (p. 224).

6. (from Jason Baldridge) Pick one (1) of the following statements and write one to two paragraphs that argues for or against it:

   (a) It would be a great idea to have a voice-activated TV.
(b) One does not need to understand what is being said in order to carry on a conversation.

(c) A dialogue system must be embodied to achieve true conversational capabilities.

(d) The Chinese room argument is right: machines will never actually be able to think because they are just manipulating symbols.

You should use concepts from the dialogue system component, such as (but not necessarily limited to) closed/open domains, depth of analysis, the uncanny valley, turn taking, and emergence.

7. (a) Take the n-gram code we wrote in class, and alter the bigram-calculating python program such that it will calculate trigrams instead of bigrams.

(b) We know that $p(w_3|w_1, w_2) = \frac{C(w_1, w_2, w_3)}{C(w_1, w_2)}$. In other words, the probability of a third word is equal to dividing the count of a trigram by the count of the starting bigram. How would you change your program in order to be able to calculate this? (You do not have to actually change your program; only describe what needs to be done.)