May 12, 2011

Contextual Vocabulary Acquisition

Inferring the meaning of the adjective "dour" from context

CSE 499: Independent Study
Spring Semester 2011
Philip Rosebrough

Abstract

In the Contextual Vocabulary Acquisition (CVA) project, we seek to understand how sentences are structured and what background knowledge is required to make deductions about an unknown word in sentences. The Semantic Networking Processing System (SNePS) is utilized to represent the ideas enclosed in a sentence containing an unknown word and combining this representation with a set of logical rules that has the ability to reach conclusions about the definition of the word. My project was to develop a semantic network for a sentence containing the adjective "dour". In order to determine the minimal background knowledge required to reach a definition of the word, human test subjects were asked to explain what information they thought was required to define the word. Nouns and verbs have algorithms that consist of general rules that apply to all nouns and verbs. However there are no known algorithms for reaching conclusions with adjectives. Therefore all deductions were produced by questions asked in my demo. (See Appendix A Page – 7) For my test sentence, the subjects that were interviewed, defined the unknown word in terms of what its antonyms are and then indirectly using derived synonyms. Future steps will include creating a database of known antonyms and synonyms of the known words, in order to globally define unknown adjectives.

1 CVA Project and SNePS

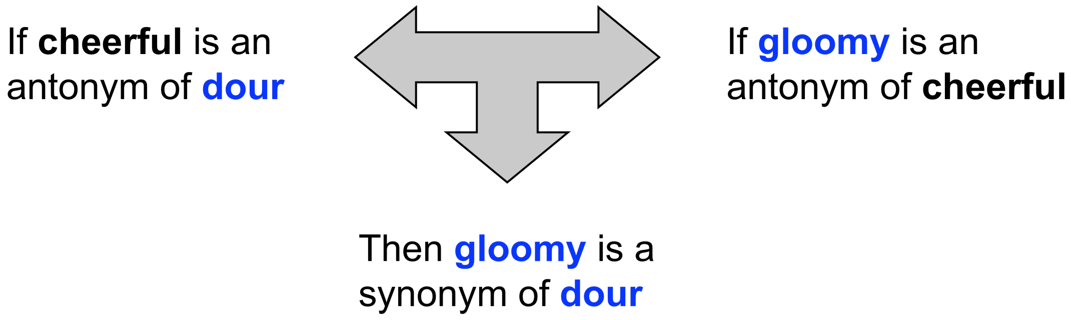
In general, humans do not look up in a dictionary every unknown word that they come across while reading. Instead we have developed ways to gather information or context from a sentence(s) that is already known to us, and then reach a logical definition, of an unknown word based on the known information. This procedure is intended to aid in the development of this skill in young students who are having trouble learning new words. By teaching a computer step by step instructions, we can efficiently teach a computer to understand information and draw "definitions" based on taught logic. Working with reading assistants and teachers, this program could be adapted and used as an efficient lesson plan to teach humans this skill.

To "teach" a computer contextual vocabulary acquisition we utilized a program known as SNePS. This program uses a graph representation to organize information, in order to successfully represent the context of a sentence. The program also has the ability to reach logical deductions based on information that is given. This given information is the rule in our sentence structures or everyday life that is followed when a person successfully infers the definition of an unknown word. By combining these two features of SNePS the program can draw logical deductions based on the sentence.

2 Research Project Role

My role in the Contextual Vocabulary Acquisition research group was to learn how to use SNePS to represent and draw conclusions about a sentence and the meaning of an unknown word in that sentence. For this project I chose the sentence "Rather than his mood of cheerful good humor, his mood appeared dour", with "dour" being the unknown word. My task was to determine how to take this sentence and deduce a definition of dour that would be equivalent to those developed by humans.

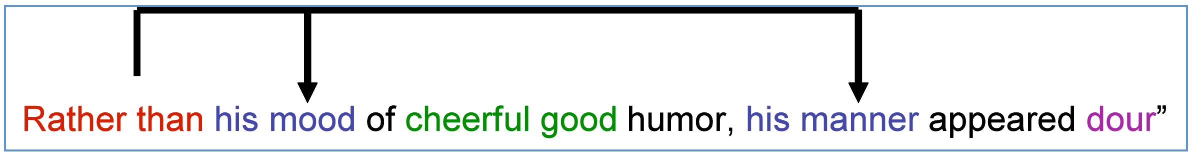
To achieve this goal I met with friends who did not know the meaning of dour; I read the sentence to the participants and asked them to infer a meaning of dour. The conclusions they drew were that, in English sentence structure when two objects are being compared by the phrase "rather than", the two objects have opposite properties which would make those properties antonyms of each other. Furthermore they reasoned that by using known antonyms of "cheerful" and "good" they could indirectly produce synonyms for "dour".


If **cheerful** is an antonym of **dour**
If **gloomy** is an antonym of **cheerful**
Then **gloomy** is a synonym of **dour**

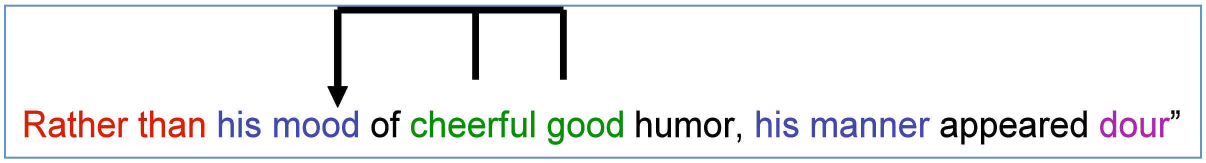
This was developed into a rule in order to mimic human behavior and logic this would be the conclusion that I expected the program to reach. The background rules were developed and converted into logical statements in my demo and were used with the represented sentence in order to deduce a definition of dour. The definition produced when asked for synonyms of "dour" was "dark", "gloomy", "and bad". The synonyms of the word in this case are specific enough to accurately define the adjective.

3 Representing the sentence

Using the SNePS program to represent the sentence "Rather than his mood of cheerful good humor, his manner appeared dour" I had to start by representing that two objects ("his mood" and "his manner") were related by the "rather than" relation.



"rather than" "his manner". Then I represented that the object "his mood" had a property of "cheerful" and another property of "good" using the object property case frame.


Rather than his mood of cheerful good humor, his manner appeared dour"

The object "his manner" was represented to have the property of "dour". It was then represented that the property "dour" was a member of the class of things that are unknown using the member class case frame. This produced the lexical representation of the sentence "his mood with the property of cheerful and good rather than his manner with the property of dour". (Appendix A Page - 7)

4 Background knowledge

In order to draw logical conclusions I needed to represent the rule that my informants used to deduce the definition of "dour". When my informants were reading the sentence, they all commented that they concentrated on the "rather than" in the sentence. They deduced from their own prior knowledge that when two objects are related by the "rather than" relation, and those two objects both have properties, those properties must be antonyms of each other. (See Appendix B.2 - Page - 8) This rule was converted into logical propositions and was used with the sentence to draw the conclusion that "dour" is an antonym of "cheerful" and "good".

I then returned to my informants and asked them what antonyms they knew for "cheerful" and "good". The list received was that antonyms were "gloomy" "dark" and "bad". A rule was created that stated that if an antonym of an object has an antonym then that second antonym is a synonym of the object. (i.e: cheerful is an antonym of dour, gloomy is an antonym of cheerful → gloomy is a synonym of dour). (See Appendix B.1 Page - 7) As a result I could ask the computer to deduce synonyms of the unknown word. This resulted in a definition of the unknown words that consisted of antonyms and then indirectly synonyms of the word. A person could then take this information and know what the word dour meant. (Appendix C Page – 8)

5 Immediate Next Steps

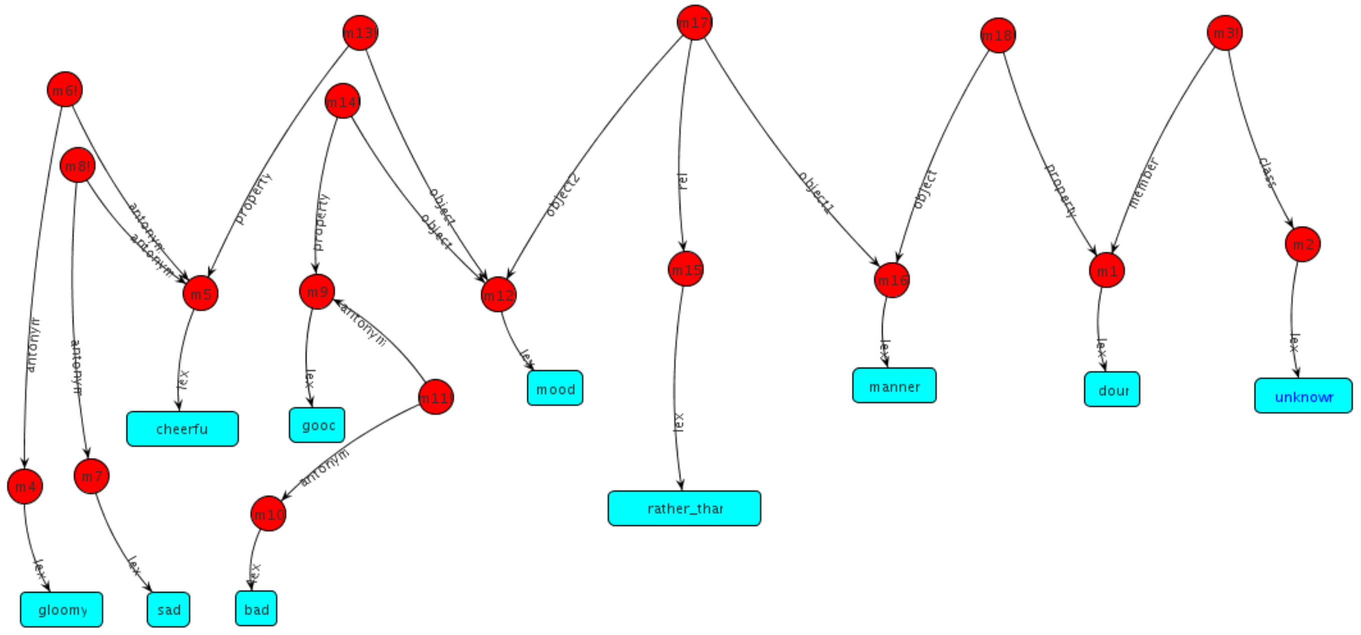
The immediate next steps to this project would be to represent the sentence more accurately. There needs to be a relation that says that the objects "mood" and "manner" are both things that are "his". This can be done with two object relative possessor case frames. Other future work is to add more known antonyms of "cheerful" and "good;" this would provide more synonyms for "dour" after deductions have been made.

6 Long Term Steps

Unlike nouns and verbs, adjectives do now have known algorithms for drawing logical conclusions. As a result of my findings, future work could consist of collecting rules that are created when students in the CVA research group work on defining unknown adjectives. With a collection of rules for adjectives, an algorithm could potentially be created to define adjectives more explicitly.

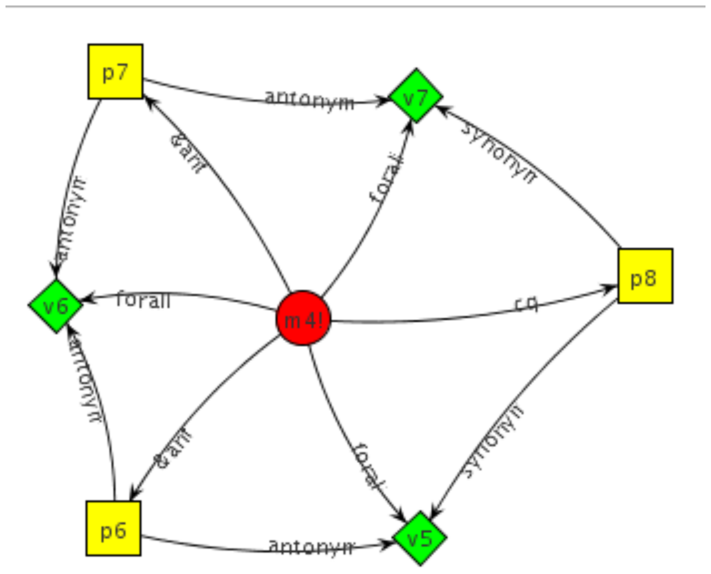
Appendix

A.) Semantic Network

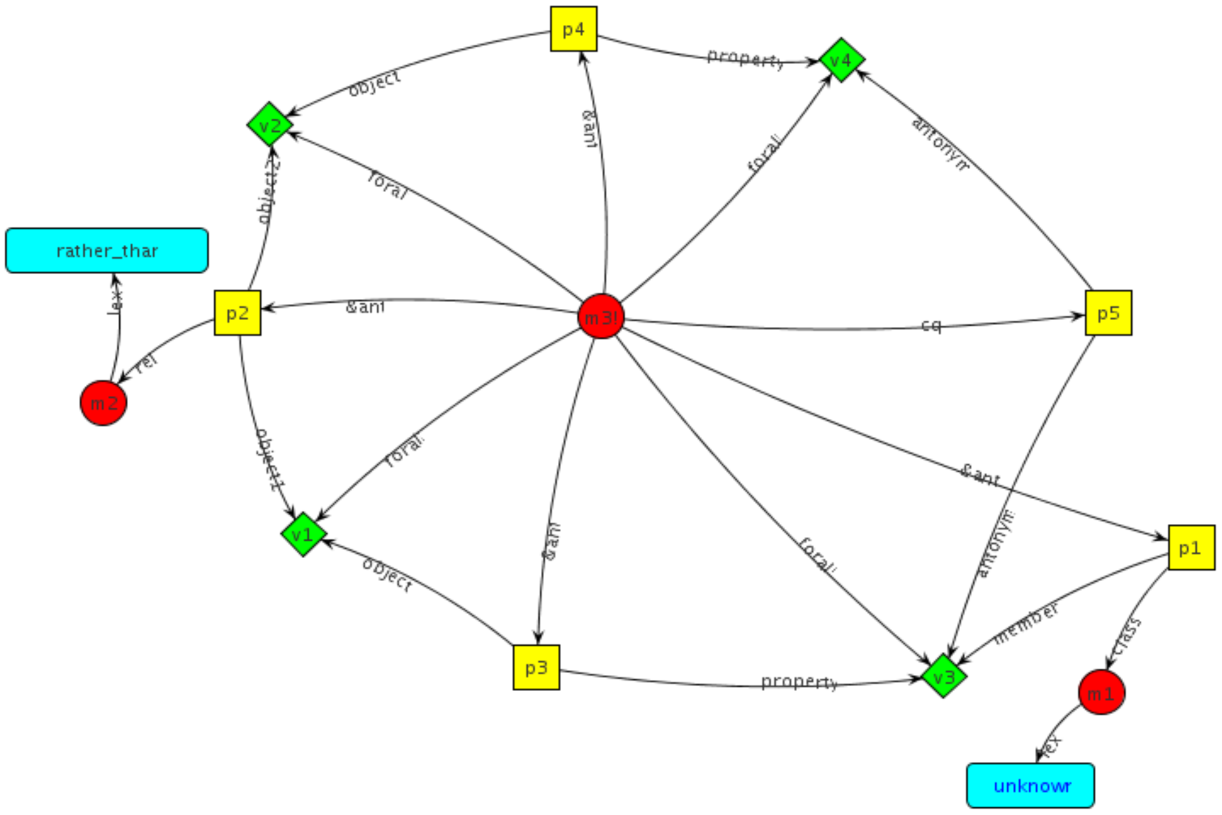


B.) Rules

B.1) If an antonym of an object has an antonym then that second antonym is a synonym of the object. (i.e: cheerful is an antonym of dour, gloomy is an antonym of cheerful → gloomy is a synonym of dour).



B.2) When two objects are related by the "rather than" relation, and those two objects both have properties, those properties must be antonyms of each other



C.) Output

```
 ; ===========================================================================

; FILENAME:      dour-demo.txt
; Date:          03-21-11
; PROGRAMMER:    Philip Rosebrough

;Antonyms of Dour;
(describe(find antonym ?x antonym(build lex dour)))

(m27! (antonym (m8 (lex cheerful)) (m5 (lex dour))))
(m28! (antonym (m16 (lex good)) (m5)))

(m27! m28!)

;Synonyms of Dour;
(describe(find synonym ?x synonym(build lex dour)))

(m29! (synonym (m7 (lex gloomy)) (m5 (lex dour))))
(m30! (synonym (m17 (lex bad)) (m5)))

(m29! m30!)
```

D.) Working demo

```
; =======================================================================
; FILENAME:     dour-demo.txt
; Date:         03-21-11
; PROGRAMMER:   Philip Rosebrough

; Lines beginning with a semi-colon are comments.
; Lines beginning with "^" are Lisp commands.
; All other lines are SNePSUL commands.
;
; To use this file: run SNePS; at the SNePS prompt (*), type:
;
;       (demo "dour-demo.txt" :av)
;
; Make sure all necessary files are in the current working directory
; or else use full path names.
; =======================================================================


; Clear the SNePS network:
(resetnet)

; OPTIONAL:
; UNCOMMENT THE FOLLOWING CODE TO TURN FULL FORWARD INFERENCING ON:
;enter the "snip" package:
 ^(in-package snip)

 ;turn on full forward inferencing:
 ^(defun broadcast-one-report (represent)
    (let (anysent)
      (do.chset (ch *OUTGOING-CHANNELS* anysent)
         (when (isopen.ch ch)
                (setq anysent
                      (or (try-to-send-report represent ch)
                          anysent)))))
    nil)
;
 ;re-enter the "sneps" package:
 ^(in-package sneps)

; load all pre-defined relations:
; NB: If "intext" causes a "nil not of expected type" error,
;     then comment-out the "intext" command and then
;          uncomment & use the load command below, instead
^(load "/projects/rapaport/CVA/STN2/demos/rels")
;(intext "/projects/rapaport/CVA/STN2/demos/rels")


; load all pre-defined path definitions:
;(intext "/projects/rapaport/CVA/mkb3.CVA/paths/paths")
^(load "/projects/rapaport/CVA/mkb3.CVA/paths/paths")


; BACKGROUND KNOWLEDGE:
; =====================
;For all w,x,y,z, if y is a member of the class "unknown", and w if related
to x by "rather_than", and w has property y, and x has property z.  Then
property y is an antonym of property z.

        (describe (assert forall ($w $x $y $z) &ant (
                (build member *y class (build lex unknown))
                (build object1 *w rel (build lex rather_than) object2 *x)
                (build object *w property *y)
                (build object *x property *z))
        cq (build antonym *y antonym *z)))



;For all x,y,z, if x is an antonym of y, and y is an antonym of z.  Then x is
a synonym of z.
        (describe(assert forall($x $y $z)
        &ant(
        (build antonym *x antonym *y)
        (build antonym *y antonym *z))
        cq(build synonym *x synonym *z)))


;"dour" is a member of the class "unknown"
(describe(add member(build lex dour) class (build lex unknown)))

;"gloomy" is an antonym of "cheerful"
(describe(add antonym(build lex gloomy) antonym(build lex cheerful)))

;"sad" is an antonym of "cheerful"
(describe(add antonym(build lex sad) antonym(build lex cheerful)))

;"bad" is an antonym of "good"
(describe(add antonym(build lex good) antonym(build lex bad)))

; CASSIE READS THE PASSAGE:
; =========================
; (put annotated SNePSUL code of the passage here)

;The object "mood" has the property of "cheerful"
(describe(add object(build lex mood) property(build lex cheerful)))

;"The object "mood" has the property of "good"
(describe(add object (build lex mood) property (build lex good)))

;The object "manner" is related to object "mood" by "rather_than"
(describe(add   rel(build   lex   rather_than)   object1(build   lex   manner)
object2(build lex mood)))

;The object "manner" has the property of "dour"
(describe(add object(build lex manner) property(build lex dour)))


;Have Cassie read in the passage inculding the word, which should
;trigger forward inference and process the definition.
;Then, ask what the word means.

;(describe(deduce synonym(build lex gloomy) synonym(build lex dour)))
;(describe(deduce synonym(build lex sad) synonym(build lex dour)))
;(describe(deduce synonym(build lex bad) synonym(build lex dour)))

;Antonyms of Dour;
```

```
(describe(find antonym ?x antonym(build lex dour)))

;Synonyms of Dour;
(describe(find synonym ?x synonym(build lex dour)))
```