Disjunctor Selection for One-Line Jokes

Jeff Stark[1], Kim Binsted[1] and Ben Bergen[2]

[1] Information and Computing Science Department, University of Hawaii
POST 317, 1680 East-West Road, Honolulu, HI USA
j-stark@hawaii.rr.com, binsted@hawaii.edu
http://www2.hawaii.edu/~binsted
[2] Linguistics Department, University of Hawaii
569 Moore Hall, 1890 East-West Road, Honolulu, HI USA
bergen@hawaii.edu
http://www2.hawaii.edu/~bergen

Abstract. Here we present a model of a subtype of one-line jokes (not puns) that describes the relationship between the *connector* (part of the set-up) and the *disjunctor* (often called the punchline). This relationship is at the heart of what makes this common type of joke humorous. We have implemented this model in a system, DisS (Disjunctor Selector), which, given a joke set-up, can select the best disjunctor from a list of alternatives. DisS agrees with human judges on the best disjunctor for one typical joke, and we are currently testing it on other jokes of the same sub-type.

1 Introduction

Here we will describe a model and implemented system that is able to solve what we will refer to as the *disjunctor selection problem* for one-line jokes. This problem is most straightforwardly introduced with an example. The following is typical of the subtype of jokes we will consider:

I asked the bartender for something cold and filled with rum, so he recommended his wife. (**1**)

Most simply, the *disjunctor* is a short piece of text, almost always at the end of the joke, that completes the text in a linguistically and logically consistent, yet unexpected and humorous, manner. In this example, the disjunctor is the phrase "his wife". The disjunctor selection task is that of providing an appropriate disjunctor, given a joke text with the disjunctor removed. That is, one must provide a phrase that will complete the text and make it humorous. Our system is designed to solve a simplified version of this problem, namely the selection of a disjunctor from a limited set of choices.

2 Background

Our basic premise is that one-line jokes, such as (1), are humorous because they violate the initial expectations of the listener, and that this violation is resolved by shifting from the initial knowledge frame used to understand the joke to another completely different knowledge frame [4]. For reasons that will need to be further investigated by psychologists, listeners often identify such violations and subsequent frame shifts as humorous. In (1), the expectation that is violated is that the bartender will fulfill the speaker's request by recommending a drink. Two factors contribute to the generation of this expectation. First, the listener initially interprets the object of the request as referring to a drink. Second, the listener makes an assumption (based on world knowledge and an understanding of how the structure of English sentences affects their semantics) that the reply to the request will involve the object of the request, *as the listener has initially interpreted it*. Hence, the listener's expectation is violated when the object of the reply turns out not to involve the object the listener had in mind.

The violation of the listener's expectations results from the semantic interplay between two elements within the joke, the disjunctor and the *connector* [1]. As previously noted, the disjunctor in (1) is the phrase "his wife". The connector is the phrase "something cold and filled with rum". The connector and disjunctor are always ultimately resolved by the listener to have the same referent, even though initially the listener assumes that the connector refers to something very different. This occurs for two basic reasons. First, the connector is semantically ambiguous (i.e. both "cold" and "filled with rum" have more than one meaning) which makes it possible for it to have more than one referent. Second, the structure of the joke demands that the connector and disjunctor have the same referent in order for the text to make sense.

In (1), the structure could be described as a request immediately followed by a reply. The connector is the object of the request, while the disjunctor is the object of the reply. In general, for most listeners, a sentence with this structure will make sense only if the object of the reply *semantically fits* with the object of the request. In other words, the two objects should relate to each other in a meaningful way. One common way two objects can be said to semantically fit is if their meanings can be resolved to the same referent.

Many one-line jokes use reference resolution to violate the listener's expectations. For example, the ultimate meaning of (1), i.e. that it conveys an insult of the bartender's wife, stems in part from the fact that the connector and the disjunctor can be resolved to the same referent. Since the connector and disjunctor refer to the same object, the properties of the connector become the properties of this object. Therefore, the bartender's wife *is* "something cold and filled with rum".

That the connector and disjunctor are resolved to refer to same object in (1), presents the listener with the problem of finding an overarching knowledge frame capable of accommodating this fact. Initially the listener attempts to understand the joke within the context of the speaker ordering a drink in a bar. This initial overarching knowledge frame doesn't fit with the final realization that the connector and disjunctor refer to the same object: after all, the bartender's wife isn't a drink. Therefore the

listener must find a new overarching knowledge frame to shift to in order to make sense of the joke. This new knowledge frame is that of the bartender insulting his wife, which semantically accommodates the disjunctor, 'his wife', having the properties attributed to the connector, i.e. 'cold and filled with rum'.

This joke is an example of an insult that plays on a negative stereotype for wives. Humor, in general, is often based on stereotypes, particularly negative ones. In (1), the disjunctor could have been 'his accountant', creating the weak joke:

I asked the bartender for something cold and filled with rum, so he recommended his accountant. (**2**)

This new 'joke' still fits our criteria, in that the listener still has to reinterpret the connector and make a frame shift in order to understand the text, but most listeners won't find it as humorous as (1), because it does not appeal to any well known negative stereotypes. Again, we leave it to psychology to answer why plays on negative stereotypes are so often at the root of humor.

Based on the theoretical considerations discussed above, we have devised the following operational model to solve the disjunctor selection problem:

1. Generate all the alternative meanings for the connector.
2. Find the negative stereotypes that semantically fit with the connector meanings.
3. Order the list of stereotype/connector-meaning pairs according to the closeness of the fit.
4. From the list of potential disjunctors, select the disjunctors that semantically fit with at least one of the stereotype/connector-meaning pairs.
5. Find the overarching knowledge frame that fits with at least one of the stereotype/connector-meaning/disjunctor triples, and which differs from the original overarching knowledge frame.

This algorithm solves the disjunctor selection problem for one-line jokes that use reference-based unification, given a finite set of possible disjunctors. Now, we will describe our implementation of this algorithm in DisS.

3 Tools

DisS is written in Prolog, because several steps in the algorithm depend on unificationn, which Prolog has built-in. Our representational formalism is based on the Embodied Construction Grammar [3]. Finally, much of the content for the basic ontology is taken from Frame Net [2].

4 System Architecture

Overall, the system is composed of two basic parts: a knowledge base and a set of operations that query or manipulate the knowledge base.

4.1 Knowledge Base

The knowledge base (KB) contains everything the system initially knows about the world. All of this knowledge is in the form of representations called *schemata*. These schemata are based on the Embodied Construction Grammar formalism, although not all aspects of that formalism are employed. Schema are frame-based representations and, as such, most of their representational power stems from organizing a set of name-value pairs called *slots* under a single reference, which is the name of the schema. Therefore all schemata have a name that is identified by the slot called **schema_name** and a set of slots, called *roles*, which are the name-value pairs that identify the attributes associated with the schema. Since this is a typical frame-based system, all the usual properties of such systems apply, such as inheritance and the overriding of ancestor slot values by descendents. However, currently none of the schemata in DisS have methods or demons associated with them.

There are two basic types of schema in DisS: *classes* and *instances*. Classes represent categories, which delineate sets of objects in the world. The most general class in the system's ontology is that of **thing**. All other schemas in the KB inherit from the **thing** schema. The objects within a class are instances of that class. Hence an instance schema is member of the category defined by some class schema in the system. Class schemas and instance schemas are differentiated by the name of the slot that indicates their inheritance relationship. Class schemas have a slot called **subcase_of**' that contains the names of the schemas they inherit from, whereas instance schemas indicate the class schemas they inherit from in a slot called **instance_of**. Below are examples of both class and instance schemas:

```
/* cold3 - property of being unfeeling or unemotional
*/
schema([
          schema_name(cold3),
          subcase_of(experiencer_subj),
          roles([ambiguous(cold)]),
          constraints([])
 ]).

/* Instance of a drink that is cold and filled with rum
*/
schema([
          schema_name(drink_inst1),
          instance_of(drink),
```

```
            roles([
                   contents([rum]),
                   temperature(cold1)
                  ]),
            constraints([])
    ]).
```

The KB contains both general world knowledge (i.e. a very basic and minimal ontology) and domain specific knowledge. The domain specific knowledge includes classes and instances necessary to represent the incomplete jokes for the DisS to process, possible disjunctors for those jokes and the stereotypes that these jokes play off of.

4.2 Representing Connectors and Disjunctors

Each incomplete joke represented in the KB is an instance of the joke schema and, as such, has a slot for the connector. The connector is itself an instance of a referent schema and has, amongst others, slots for the category of the referent and its attributions. Each part of the connector that is semantically ambiguous is represented by making it an argument to a predicate called **ambiguous**. So since the word "cold" in "something cold and filled with rum" is ambiguous, it is represented as ambiguous(cold). The phrase "filled with rum" is handled in a similar manner. The connector schema for (1) therefore, looks like the following:

```
            schema([
              schema_name(ref_inst_thing),
              instance_of(referent),
              roles([
                     category(thing),
                     restrictions([]),
                      attributions([ambiguous(cold),
                         ambiguous(filled_with_rum)]),
                     number(singular),
                     accessibility([]),
                     resolved_ref([])
                   ]),
              constraints([])
            ]).
```

Based on this knowledge, the system is able to generate alternative meanings by creating a set of new referent schema each with a different combination of meanings for each of the ambiguous parts of the connector. The KB contains three distinct meanings for "cold" and two for "filled with rum", therefore the total number of possible meanings for the connector "something cold and filled with rum" is six. Note

that all possible meanings are generated whether or not they make sense. Also note that the generated meanings are not semantically ambiguous.

Once all the possible meanings of the connector have been generated, the system next tries to semantically fit each with a stereotype in the KB. Each stereotype has a slot, **applies_to_category** that indicates which category it is a stereotype of. The remaining slots of the stereotype identify the typical attributes associated with it. Below is an example of the bad wife stereotype as it is represented in the KB:

```
schema([
  schema_name(bad_wife_stereotype),
  subcase_of(stereotype),
  roles([
        applies_to_category(wife),
        emotional_disposition([cold3]),
        cognitive_disposition([intoxication]),
        moral_disposition([unfaithful])
      ]),
  constraints([])
]).
```

Matching alternative meanings to stereotypes is a two-step process. First, an attempt is made to semantically fit the category of a stereotype with that of the meaning currently being considered. A stereotype and a meaning are considered to fit if they are identical, or if one subsumes the other. If there is a match, then an attempt is made to fit the values of the remaining slots of the stereotype with those of the attributions of the meaning being considered. Again, a stereotype attribution and an alternative meaning attribution fit if they are identical, or if one subsumes the other. A count is made of the number of meaning attributions that fit with stereotype slot values as well as those that do not. The degree of how well a stereotype fits with a possible meaning is indicated by the difference between the number of meaning attributions that matched and the number of those that did not. Only those stereotypes that *both* fit with the category of the possible meaning *and* have a positive degree of attribution fit are included in a list of best fitting stereotypes for that meaning.

As an example, consider matching the stereotype above with the generated meaning represented below:

```
schema([
  schema_name(ref_inst_thing7),
  instance_of(referent),
  roles([
         category(thing),
         restrictions([]),
          attributions([cold3,
                        filled_with_rum2]),
         number(singular),
```

```
                accessibility([]),
                resolved_ref([])
            ]),
        constraints([])
    ]).
```

The schema called **cold** represents the definition of the word "cold" that means unemotional. Likewise the schema called **filled_with_rum2** represents the meaning of the phrase "filled with rum" that means to be in a state of intoxication. Since the bad wife stereotype applies to the category **wife** which is subsumed by the category **thing**, the **ref_inst_thing7** passes the first matching test with **bad_wife_stereotype**. The degree of the match is two, because one of the attributions (**cold3**) of **ref_inst_thing7** is identical with a slot value in **bad_wife_stereotype**, and the other attribution (**filled_with_rum2**) is subsumed by a slot value in **bad_wife_stereotype** (**intoxication**). Therefore the bad wife stereotype would be placed on a list of best fitting stereotypes for the referent.

Incomplete jokes (i.e. missing a disjunctor) are also represented in the KB. The schema representing (1) without its disjunctor is called **joke_instance1** and is shown below:

```
schema([
        schema_name(joke_instance1),
        instance_of(joke),
        roles([
               basic_semantics(req_reply_inst),
               connector(ref_inst_thing),
               expectation(drink_inst1),
               inital_context(ordering_a_drink_in_bar)
              ]),
        constraints([])
 ]).
```

5 Results

When operating on **joke_instance1**, the system is able to a) generate all the possible meanings of the connector (i.e. "something cold and filled with rum") and b) for each possible meaning generate a list of those stereotypes in the KB that meet the criteria mentioned above. Table 1 shows the output generated by the system when operating on **joke_instance1** (note that only matches with at least one attribution match are shown).

Table 1. Matching alternative meanings of the connector with negative stereotypes

Meaning of "cold"	**Meaning of "filled with rum"**	**Matching stereotypes (degree of attribution fit)**
Having a low temperature	Containing rum	None
Having a low temperature	Intoxicated with rum	Bad wife (-1+1=0)
Feeling chilled	Containing rum	None
Feeling chilled	Intoxicated with rum	Bad wife (-1+1=0)
unfeeling, unemotional	Containing rum	Bad wife (1+-1=0), accountant (1+-1=0)
Unfeeling, unemotional	Intoxicated with rum	Bad wife (1+1=2), accountant (1+-1=0)

The system's output listed above makes clear that of all the possible meanings generated for the connector the one that has the best fit with a stereotype in the KB is the one that means "something unemotional and intoxicated". The best fitting stereotype in this case is the bad wife stereotype, which only applies to things that are members of the **wife** category. According to our model, disjunctor must ultimately fit with one of the possible meanings of the connector and, in order to generate humor, a negative stereotype that fits with this possible meaning. Selecting as an alternative meaning for the connector the referent that means "something unemotional and intoxicated" will impose as a selection criterion for the disjunctor that it fit with the **wife** category. Therefore, only those possible disjunctors represented in the KB that fit with the **wife** category could be reasonably considered by the system. The referent schema that represents the bartender's wife, **bartenders_wife** exists in the KB,[1] and is an instance of the category **wife**, so there is a fit.

There is still one step required to show that **bartenders_wife** (or rather, an English phrase referring to this schema) would make a good choice as disjunctor. The system must still satisfy the major theoretical premise upon which the system rests, namely that jokes force a frame shift. What the system needs to do is show that the alternative meaning of the connector and the selected disjunctor fit within a frame that captures the overall meaning of the joke, and differs from the frame supported by the set-up (i.e. **ordering_a_drink_in_bar**). Luckily, such a frame exists in the KB, namely the **insulting_someone** schema, which requires that someone either directly connected to someone in the scene (e.g. married to the bartender) or generally well-known be identified with a negative stereotype.

[1] It was beyond the scope of this project to build a system able to infer the existence and attributes of the bartender's wife, so a suitable schema was hard-coded into the KB.

So, DisS is able to choose **bartenders_wife** as the best disjunctor for **joke_instance1** from a list of alternative disjunctors, shown in Table 2. All human judges asked to choose from the same list of disjunctors made the same choice.[2]

Table 2. Alternative disjunctors for joke_instance1

Disjunctor schema	Disjunctor text	Suitable disjunctor?
bartenders_wife	"his wife"	Yes
bartenders_accountant	"his accountant"	No: degree of match too low
mai_tai	"a Mai Tai"	No: no negative stereotype, no frame shift
accountants_wife	"an accountant's wife"	No: not known or connected to scene, so no frame shift
basketball	"a basketball"	No: doesn't fit with alternative meaning

The system should of course also be demonstrated on more than one joke.[3] We are currently in the process of adding schemata to the KB to support processing of the following joke:

I asked the bartender for the quickest way to the hospital, so he recommended a triple martini. (**3**)

This is based on the rather more famous joke:

I asked the lady how to get to Carnegie Hall, and she said "Practice, practice, practice!" (**4**)

6 Conclusion

We have described an operational model of the connector/disjunctor relationship in one type of one-line non-punning joke, and implemented a system, DisS, which can

[2] So far, we have asked five fluent English speakers somewhat informally. We plan to complete a more formal evaluation in time for INTETAIN 2005.

[3] We plan to have two more jokes implemented in time for INTETAIN 2005.

choose the best from a list of possible disjunctors for a joke set-up. DisS has been shown to agree with human judges on one typical joke, and we are currently testing it on other jokes of the same type.

References

1. Salvatore Attardo. 1994. Linguistic Theories of Humor. Mouton de Gruyter, Berlin and New York.
2. Collin Baker, Charles Fillmore, and John Lowe. 1998. The Berkeley FrameNet project. In Proceedings of the COLING-ACL 1998, Montreal, Canada.
3. Benjamin K. Bergen and Nancy C. Chang. 2002. Embodied Construction Grammar in Simulation-Based Language Understanding. Technical Report TR-02-004, International Computer Science Institute, Berkeley.
4. Seana Coulson. 2001. Semantic Leaps: Frame-shifting and Conceptual Blending in Meaning Construction. Cambridge University Press, Cambridge, U.K. and New York.