Semantic Clustering

Acquisition of Partial Ontologies From Public Domain Lexical Sources:

First Experiments

Cao Feng, Terry Copeck, Stan Szpakowicz, Stan Matwin

Department of Computer Science, University of Ottawa
Ottawa, Ontario, Canada K1N 6N5
{cfeng, terry, szpak,stan}@csi.uottawa.ca

Abstract

In recent years a number of machine-readable natural language resources have become available in the public domain. These include text corpora, lexicons and lexical networks. We document our efforts to use public resources in a variety of ways on projects which acquire knowledge from technical text and perform machine learning on the resulting representations. The practical issues involved in one task employing a public resource, the construction of semantic clusters of concepts, are explored in greater detail.

1. Introduction

This paper is meant to serve two purposes. It summarizes the current status of two projects underway in the research laboratories of the University of Ottawa's Department of Computer Science. The discussion places particular emphasis on identifying how two public domain Natural Language resources have been integrated into the projects. We then look more closely at one particular task, clustering concepts which have been extracted from a text according to the relationships encoded in a public lexical network.

At the moment a staff of four graduate students, three postdoctoral fellows, two faculty and a A.I. programmer are actively working on the two projects. Both projects are partly under design and partly implemented, and we are developing a prototype to test our ideas. We are aware that these ideas are quite ambitious and view our work as exploratory rather than definitive. The architecture detailed below is therefore likely to change in future, perhaps quite drastically.

2. Background

The two projects in our lab in which knowledge is acquired from text are TANKA and MaLTe. Both use the Collins and WordNet public domain resources we have incorporated in our designs in the last half year. Figure 1 and the descriptions that follow introduce the top level architecture of both projects and discuss how they rely on public domain lexical resources for a variety of purposes, in particular the initial partial acquisition of basic ontologies.

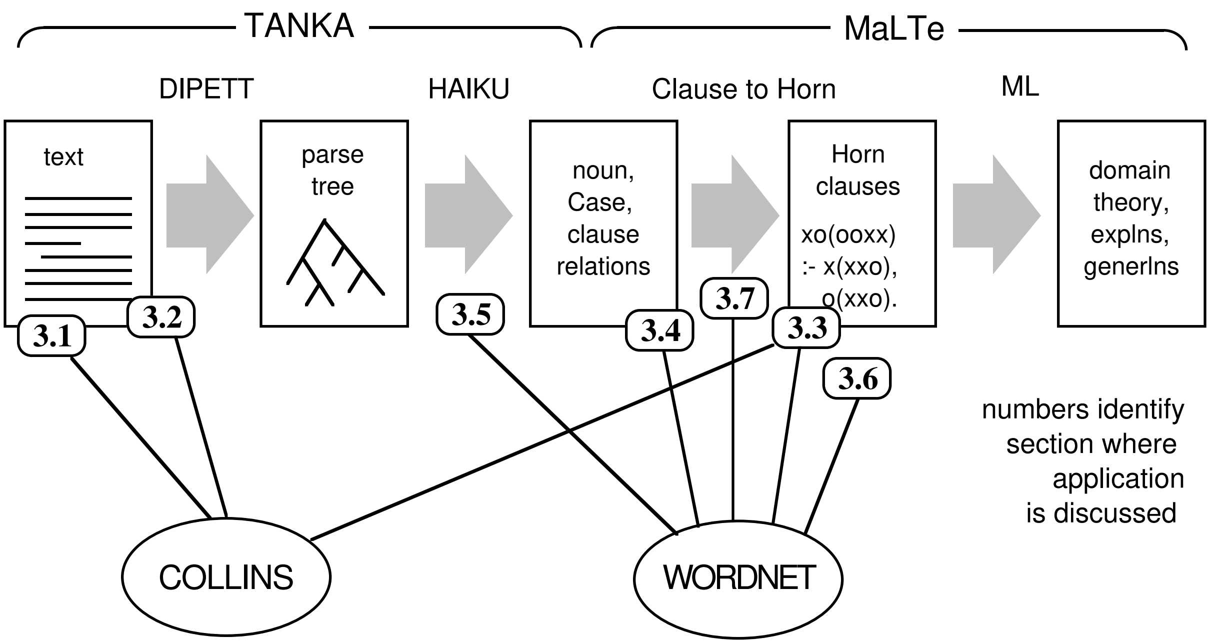

Figure 1. Components of Current Work

2.1 TANKA

Text ANalysis for Knowledge Acquisition is the older of the two projects in which we are engaged (Delisle, Szpakowicz, Copeck & Barker, 1992; Copeck, Delisle & Szpakowicz, 1992; Delisle & Szpakowicz, 1991). TANKA seeks to build a model of a technical domain by semi-automatically processing unedited English text that describes the domain. In doing so it employs a classification of the linguistic realizations of semantics into a hierarchy of three types[1]. Nouns and their modifiers constitute its lowest level. Noun phrases typically fill slots in the Case relations surrounding verbs. Verbs together with their frame of Cases—clauses—constitute the intermediate level of the hierarchy. Relations between clauses form the top level of our linguistic-semantic hierarchy.

TANKA's processing consists of parsing sentences and extracting from the parse trees patterns of the words that mark Cases for each verb. Each of these Case Markers is mapped to the Case it realizes. The resulting Case Patterns are assembled into network fragments using an indigenous but typical formalism. These fragments, merged into a single network, constitute the Conceptual Network of the domain as communicated by the text.

TANKA is composed of three modules: the fully-implemented DIPETT parser (Domain-Independent Parser for Technical Texts; Delisle & Szpakowicz, 1991), an almost fully implemented Case semantic analyzer called HAIKU (Delisle, Barker, Copeck, & Szpakowicz, 1992), and a network fragment builder (MNB: Mini-Network Builder) currently in design. Other interesting features of this system are its minimal reliance on pre-coded knowledge in favor of repeated recourse to the user for selection or confirmation and its attempts to learn from this user input. TANKA's objective is to learn to provide the user with the correct options in each situation.

2.2 MaLTe

The department's Machine Learning group are interested in learning directly from natural language texts. A text communicating a set of rules provides a suitable subject for machine learning when its narrative is interspersed with examples[2]. In ML terms, the narrative provides the domain theory and the examples constitute the training set.

The Machine Learning from Text project (Matwin & Szpakowicz, 1992, 1993; Delannoy, Feng, Matwin & Szpakowicz, 1993) is the result of this observation. We hypothesize that machine learning will produce deeper knowledge of a domain than that produced by TANKA' s processing of the literal text. In practical terms, ML will invent new concepts or links consistent with the Conceptual Network which do not appear in it because they were not explicitly stated in the text. These abstractions or generalizations can then be passed back to TANKA for incorporation in the domain representation.

MaLTe's ML module applies both traditional ML approaches like EBL and modifications to techniques like absorption and ILP. These processes and the problems attendant on their application to Horn clause representations of a text are already well understood and work on this module using hand-coded input is well advanced.

What is more difficult is formulating the operations necessary to convert textual representations, whether parse trees, Case Patterns, or mini networks, into the sequence of linked and fully specified Horn clauses required for machine learning. The Case-to-Horn (CtH) module in this project is the most speculative and least advanced of our various undertakings. It is also arguably the most interesting and the one requiring the greatest assistance from external sources. Moreover, we will rely on machine learning to decrease the amount of user assistance as the use of the system progresses.

Some of the tasks CtH must address are distinguishing between new or existing entities, dealing with sets and with numerical values, and inferring implicit temporal or causal relationships. These are difficult tasks, and addressing them definitely requires knowledge. A tenet of our approach is to minimize the need for acquiring knowledge necessary to perform text analysis. Consequently, we have turned to public domain sources from which this knowledge can be obtained semi-automatically. To see specifically what task we are addressing here, consider the phrases 'take care of' and 'look after'. These word sequences must first be recognized as unitary phrases and then as synonyms or near synonyms. The human reader does this almost automatically. Not so for the computer system, which can only draw such conclusions when evidence warrants—and in our case, precoding information is ruled out. It is not surprising that along with an involvement with text, MaLTe inherits from TANKA the need to interact with the user to clarify these ill-defined areas.

2.3 The Collins Dictionary

The Collins Dictionary is a 9 megabyte list of 311,000 words classified according to nine POS[3] {`N, V, PropN, A, Adv, Pron, Prep, Conj, Det`}. Each POS in an entry is followed by up to three qualifiers which specify its particulars. Words may be classified as more than one POS (`'outpouring outpouring N sg#outpour V PROG'`) or more than one form of a single POS (`'outclassed outclass V PAST WK#outclass V PPART WK'`), a sharp symbol is then used to delimit each definition. The Collins flat file represents inflected forms distinctly and its inflections include punctuation (`'outpost, outpost's, outposts, outposts'`). Each definition includes the root form of the word. There are around 90,000 root forms in the file.

The 1979 Collins dictionary came to us as part of the PC-KIMMO two-level morphological processor for English implemented at the University of Pennsylvania (Karp, Schabes, Zaidel and Egedi, 1990). Although PC-KIMMO includes programs to construct a 19 megabyte lexicon in memory from the dictionary flat file, we found it more useful to work directly with the flat file from within Quintus Prolog.

Our first step was to sort Collins into ascending alphanumeric order[4]. Although the ASCII collating sequence differs from standard lexicographic order in its separation of upper and lower case, this posed us no difficulties— for our purposes first-letter-capitalized entries not recognized when fully lower case can be taken as proper nouns in almost all cases. We then wrote Prolog functions to look up words in the sorted file by binary search; although not as fast as hashed access this was adequate for our purposes. In addition to supporting binary search, the sorted file allowed us to retrieve all inflected forms of the same root word from its neighborhood efficiently. This operation is necessary when searching for a word's derivational counterparts.

2.4 WordNet

This multiply-linked network of lexical items takes advantage of the computer to construct a network of semantic relations which is not limited to the sequential ordering imposed on words on a page. The fundamental unit in WordNet is the synset, a grouping of words and phrases which together specify a distinguishable sense. Synsets can be linked to one another along twelve relational axes. WordNet is the product of a working group at Princeton University under the direction of George Miller (Miller, Beckwith, Feilbaum, Gross & Miller, 1990). First publicized in late 1990, version 1.3 of the system has been ported from Unix to the NeXT, DOS and Macintosh platforms.[5]

Of the twelve kinds of relationships possible between synsets, half are based on lexical or syntactic associations: 1) polysemy or familiarity, *'looks like'*; 2) collocation, 'occurs *with'*; 3) (verb) frame, *'goes with'*; 4) (adjective, adverb) derivation, *'pertains to'*; 5) synonymy, *'is like'*; and 6) antonymy, *'is unlike'*. The other six links express purely semantic relationships. Three are types of the holonymic or compositional relationship involving nouns alone: 7) *'part of'*; 8) *'substance of'*; and 9) *'member of'*. A fourth, 10) etiology, *'implies',* inks verbs only, and the last two connect both nouns and verbs: 11) hyponymy, *'is a'*; and 12) fraternity, *'is sibling'*.

Although primarily semantic, WordNet does incorporate a limited amount of syntactic information. It recognizes four disjoint syntactic types (nouns, verbs, adjective and adverbs). According to 1990 data[6], WordNet contains 33,400 noun forms gathered into 26,300 synsets. Its 12,000 adjectives define 10,500 synsets. 8,000 verb forms express 4,700 synsets. In total, then, this network records about 53,000 words and 41,500 semantic senses.

We are fortunate in being able to use WordNet as received, issuing commands to it and receiving results via Quintus Prolog's Unix operating system interface. WordNet output is text formatted for display, so a major and wholly uninteresting effort on our part has been to write translators to recognize the format conventions and convert output into terms Prolog can operate on. A second task has been to bridge the barriers between the different syntactic classes. Being able to move from a verb form ('attach') to a noun ('attachment') will increase our ability to find links between disparate concepts in our representations. At present we are using three elements of WordNet: the memberships of synsets, records of collocations, and the hypernymy/hyponymy relationship.

3. Uses

Our projects use Collins and WordNet in seven distinct ways. All three of the Collins uses involve word morphology or syntax. WordNet participates in four other functions, one of which is lexically-oriented and the other three concerned with semantics. The different ways are described below.

3.1 Supporting Interactive Definitions of New Words

Our perspective on written text views much of it as directed to all members of the community of language users. Fiction is a good example. Narratives of this sort employ common knowledge and commonsense and this makes them difficult to process automatically. We also however distinguish specialized writing such as technical text which, because of its more limited scope and assumptions, is more amenable to our sort of mechanical processing. This benefit comes with a certain cost—each technical domain introduces its own set of terms and meanings which must be acquired for processing to occur.

When DIPETT finds a word it does not recognize it asks its user to identify the new word's POS. Depending on the selection made other questions are put to the user to complete the definition. Although users quickly become versed in the details of the DIPETT grammar, they often find it useful to see any Collins entry for the word's syntax and any WordNet entry for its semantics before competing the definition.

3.2 Generating New Word POS Definitions Automatically

The interaction between system and user described above requires the user's constant attendance at the computer. This may not always be possible so we have implemented a batch mode[7] in which DIPETT automatically adopts Collins entries for words it does not find in its own dictionary. Because Collins definitions are less specific[8] than its own, this requires making some assumptions about whether for instance an unknown noun is a count noun or a mass noun. Although this may require more attention at some future date, automating new word definition has freed us to turn our attention to other areas of the project.

3.3 Searching for Lexical Peers

Machine Learning can only occur when the facts being reasoned on are linked to one another sufficiently for inferences to be made. The expression of such links in text sometimes crosses POS. 'Attach the forms' may subsequently be referred to as 'the attachment'. To follow such a step we must be able to recognize the connection between noun and verb forms of a given root word. For ease of reference we term these derivational counterparts *peers.*

WordNet's derivation relationship, *pertains to*, links adjectives and adverbs with the nouns and verbs with which they are associated. This is particularly useful when the modifier is based on a different root than the word it touches on. However WordNet only records these two POS associations and does not include links between inflected siblings.

We have written a program supplementing WordNet for noun→verb and verb→noun associations. The program takes advantage of the alphabetic ordering of the Collins file. If the given word is found in the file, 25 words before and 25 after its entry are loaded into memory (the presumption is that all inflected peers will be found within this vicinity). The given word is then decomposed into a root and nominalizing suffix. This root and alternate forms generated by appending verbalizing suffixes are sought in memory. The first form found and its definition are returned. This program can be backtracked into if the first form is unsatisfactory. The lists of suffixes to convert one POS to another were taken from Fromkin & Rodman (1988) and Quirk, Greenbaum, Leec h and Svartvik (1985).

An example may clarify thie process. If we ask the program to find nouns related to the verb `attached`, it looks this inflected form up in Collins and returns its root, `attach` (another presumption is that this roots is the basis for all derived forms, no matter what POS). The program then appends each of the nominalizing suffixes found in the `verb_noun/1` database below. `Attach`, `attachage` `attachan` etc are all sought unsuccessfully in Collins. `Attacher` is the first word for which a dictionary entry exists. If it is unsatisfactory, `attachment` will be found next.

```
noun_adj("ish","ous","an","esque","ate","ful","ic","like").
noun_verb("","ize","ate","ish").
verb_noun("","age","al","ance","ant","ation","dom","ee","ence","er","or","ist",
          "ing","ion","ment").
adj_verb("th","en").
adj_adv("ly").
adj_noun("ity","ness").
```

Table 2. Suffixes to Change Part-of-Speech

3.4 Recognizing Phrases

In addition to idioms [9] our grammar recognizes a large class of compound nouns and phrasal verbs whose meanings are only slightly altered from those of their constituent words. The most important hallmark of '*soup pot*', '*look up*' and '*take care of*' is that they should in most cases[10] be treated as single units rather than as individual words.

The difficulty both with idioms and phrases is recognizing them. A number are stored in DIPETT's dictionary, but a greater number are not. Collins is of no use to us here; all of its entries are for single words. However a respectable number of the entries that define WordNet synsets are precisely the sort of phrase we are interested in. While failing to find a phrase in WordNet does not necessarily rule out its being one, success is a strong indication that we can treat it as a single item.

3.5 Establishing a Hierarchy of Semantic Categories

To deepen the representation and to increase the system's power to reason over concepts, we intend to record semantic labels for the nouns that fill Case roles and possibly for other POS. Selecting a set of labels is not a simple issue. When we investigated the issue a year ago we found a number of candidate sets, many of which shared a great deal yet differed in significant ways. Incorporating WordNet into our projects resolves the issue for now. Pursued to its root, WordNet's hypernym link for nouns identifies ten disjoint branches of *is-a* hierarchy. If we assume that these constitute the first level below an unstated root ('thing, entity' ?) then the first three levels of WordNet classify nouns into 94 classes (Appendix A). A similar taxonomy can be developed for verbs.

Once we have settled on a semantic categorial system, we have a framework around which to build clusters of the concepts communicated by the texts we process. Because the system is a hierarchy, the possibility exists of generalizing from particular to more generic notions. Because the system is unitary, our clusters and generalizations can be consistent with one another. In this way we hope to structure our representation of the knowledge in a domain.

3.6 Providing a Context for Measuring Semantic Affinity

In addition to tracking concepts across POS boundaries and devising a system of standard semantic categories, our work would be assisted by a general capacity to measure semantic similarity. The number of steps, link traversals, between two synsets is an obvious measure. Words belonging to the same synset have 0 distance; they are synonyms, and identifying synonyms is so evidently a use of WordNet as almost not to require mention. Moving up the hypernym and down the hyponymy relationship allows us to connect words which are not synonyms but have some semantic similarity. We expect this will be an important use of WordNet as we develop the CtH module.

3.7 Disambiguating Word Senses and Semantic Clustering

Perhaps the most obvious purpose of WordNet is to enumerate the different meanings a word or phrase can express. To process a text's meaning, we must select among different possible senses in the same way that DIPETT distinguishes which of the different POS a word can be when it parses a sentence. This is done almost unconsciously by human readers but is difficult to automate. Words which have a single sense are easy. When words have multiple senses we are using the heuristic—the most likely sense is the one which occurs most frequently in the text. In this way a word and its synonyms can be used together to single out their common sense.

However, before this can be done we must have a capacity to identify synonyms or *semantic clusters*. WordNet helps here; its fundamental unit is the *synset*, or set of synonyms, which define a semantic sense. Not all synonyms are present in WordNet, though, and those which are absent are not easily identified. A program which undertakes this task is described in greater detail below.

4. Semantic Clustering Examined More Closely

Natural language is replete with words and phrases that can express the same meaning. Despite its relative simplicity, this remains a problem in technical text. A substantial body of knowledge is required to associate phrases of similar meaning and to distinguish phrases whose meanings differ. Knowledge of this sort is tied to its domain ('domain knowledge'), however, and takes a good deal of labor to engineer. For these reasons we have avoided a knowledge-intensive approach in favor of one which relies on the system user to supply knowledge as needed—an interaction-intensive approach.

Frequent dialogues run the risk of irritating or exhausting a user. To minimize this we want to make maximum use of public domain resources and limit user interaction as far as possible to confirming or selecting among the options they suggest. Although from the point of view of MaLTe these resources could have been introduced at the ML stage (Figure 1), a number of reasons led us to decide that this would be too late[11]. Accessing large files repeatedly needs no user interaction but is time consuming, so we removed the program to do it from the interactive portion of the system. Running it once off-line generates all required word sense associations prior to interaction with the user.

To illustrate the need for this undertaking, consider this extract from the <u>Supplement to Child Care Expense Claims of the 1990 Canadian Tax Guide</u>:

```
Child care expenses are amounts you pay to someone to look after your children. T
include amounts you pay to:
   •  friends or neighbours, or other persons,
   •  relatives 21 years of age or older, …
```

It defines the concept of `child care expenses`. However it does so using imprecise natural language concepts such as `'look after'`, `'other persons'` and `'relatives'`. These phrases must all appear in predicate names to axiomatize the above narrative.

Compare it with another portion of the text giving an example of the concept:

```
Jim, a member of the Canadian Armed Forces, was posted to Lahr, West Germany in 19
While in Lahr, … Jim … paid $1,500 to a German neighbour to take care of their chi
These payments qualify as eligible child care expenses.
```

Note the different use of `'take care of'` and `'look after'`. Although both phrases express the same semantic concept and should produce the same axiomatization, they are not exactly the same lexically. They need to be associated, or *clustered*, with each other through some semantic means.

We have found in WordNet a convenient means to make this association. A program to use this public domain resource has been implemented in Quintus Prolog. It uses the Collins dictionary n addition to WordNet's synsets. The program is composed of the five phases illustrated in Figure 2. We will discuss each separately.

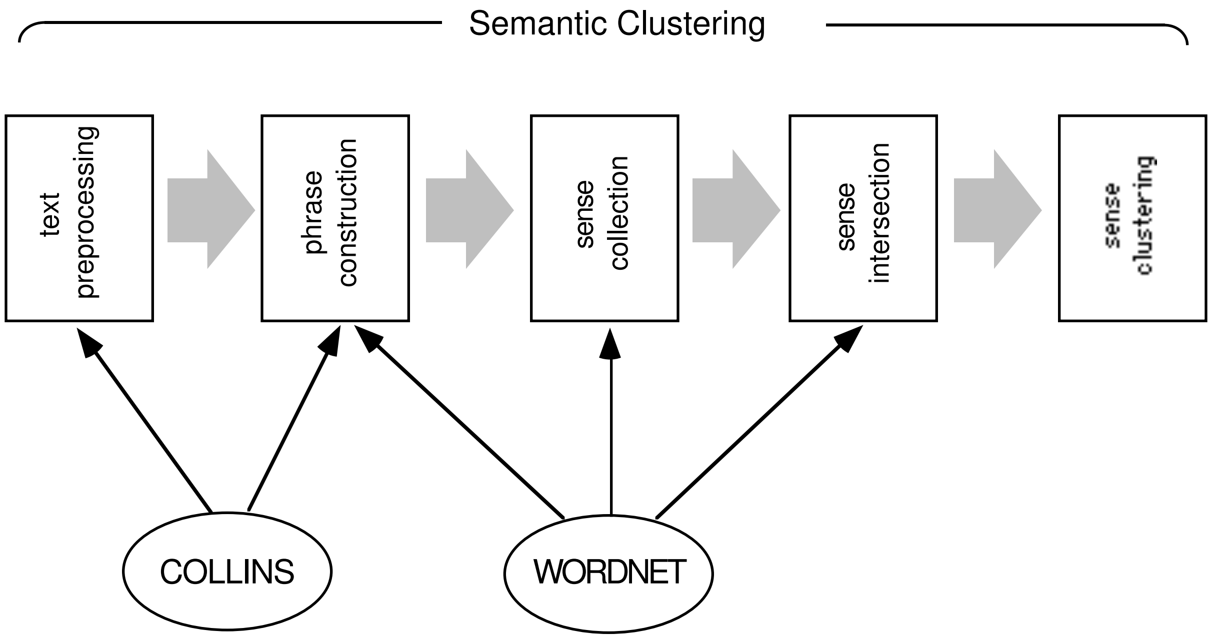

Figure 2. Five Stages of Sense Clustering

4.1. Preprocessing the text

The first task was to normalize the sentence by reducing each inflected word to a root form. Plural nouns forms are converted to singular and verb tense and number removed with the aid of the Collins dictionary. A more difficult task was to identify each word's POS, based not only on its dictionary definition but also on its location in the parse tree. Six POS are recognized: verbs, nouns, adjectives, adverbs, prepositions and others. Implementing this phase involved extending the original DIPETT dictionary slightly.

4.2. Recognizing phrases

The next phase in semantic clustering is concerned with recognizing phrases in the input sentences which should be treated as single units. The input to this phase is a set of tagged sentences, each of which contains the number of the sentence in the sequence and the sentence itself, together with a list of singular words tagged with their individual POSs. The following shows the first sentence in the child care tax guide text:

```
sent( 1,'[what,are,child,care,expenses,?]'% sentence and number
         [tag(what, other),                 % tagged words
          tag(are, verb),
          tag(child, noun),
          tag(care, noun),
          tag(expense, noun),
          tag(?,other)  ]).
```

Phrase recognition operates by taking a sequence of words, from the tagged list. The size of this sequence is fixed; in most cases five is sufficient. WordNet is then searched for an entry matching the phrase[12] which these words form. If none is found the sequence is reduced by dropping its last word and searching again. Any matched phrase replaces the words that compose it in the tagged list. The program then continues from the next unprocessed word until the list is exhausted.

Accessing WordNet is time consuming and should be minimized even in a batch process. To do this we restrict the POS that can form a phrase. POS allowed in noun phrases are: 1) noun, 2) noun + noun phrase, 3) adjective + noun phrase. Verb phrases can contain: 1) verb, 2) verb + adverb (+ preposition), 3) verb + noun + preposition.

The results of recognizing phrases in the example sentence is:

```
sent( 1, '[what,are,child,care,expenses,?],' % sentence and number
          [tag(what, other),                    % tagged words
           tag(are, verb),
           tag('child care', noun),
           tag(expense, noun),
           tag(?, other)  ]).
```

Note that the two nouns `child` and `care` have been folded into the noun phrase `'child care'`. The sequence of words `take`, `care`, and `of` would fold into the verb phrase `'take care of'` similarly.

4.3. Batch processing word senses

Next all senses of words and phrases in the tagged list are retrieved from WordNet and stored in a `sense` representation. The representation identifies the number(s) of all sentences in the text where the word or phrase appears, lists all of its senses, and lists any similar words or phrases found elsewhere in the text. At this point in the process this last list is empty. Here is the word sense entry for `'take care of'`:

```
sense('take care of', verb,         % word or phrase and POS
      [40,5],                              % #s of sentences where it appears
      [ ],                                 % reserved
      [                                    % list of senses
      ( [manage, 'deal with', handle, 'take care of', mind, coordinate, watch]
       -[control, command, 'have control over', 'have power over']) ]).
```

This verb has a single sense. The first item in its representation is the phrase itself followed by its POS. Next is a list of the sequence number(s) of the one or more sentences where it appeared. This is followed by a list of similar words or phrases. The final item is a list of the word or phrase's senses. Its contents require some explanation. WordNet distinguishes two sorts of word senses, those that are similar and those that are related (less similar), and separates them by the symbol '-'.

Although we have abided by this convention, we treat both groups the same when searching for semantic clusters.

4.4 . Intersecting similar word senses

Senses are now intersected to identify the words and phrases which share them. This information has many uses. For instance, similar words and phrases are likely to be converted into the same predicates in the CtH process. Words or phrases similar to a given sense are recorded in the list that forms its fourth element. The sense for `'take care of'` looks like this after the intersection phase:

```
sense('take care of', verb,          % word or phrase and its POS
      [40,5],                              % #s of sentences where it appears
      ['look after' -[watch] ]       % similar phrase and sense
      [                                    % list of senses
      ([manage, 'deal with', handle, 'take care of', mind, coordinate, watch]
      -[control, command, 'have control over', 'have power over']) ]).
```

We can associate the meaning of `'take care of'` and `'look after'` closely in many contexts[13] . As expected, WordNet relates the former to the latter through the sense `watch`. The above case is very simple. In practice a word or phrase may be similar to several others in the text and via several different senses. We look at a more complex example:

```
sense('look after', verb,            % word or phrase and its POS
      [50, 2],                             % #s of sentences where it appears
      [bear -[hold],                       % similar phrase and sense
       care for -['care for', 'give care to', 'provide care for'],
       maintain -['have charge of', hold, keep, 'look after'],
      'take care of' -[watch] ]
      [                                    % list of senses
      (['attend to', 'provide for' , 'look after' ]
      -['care for', 'give care to', 'provide care for']),
      ([keep, 'have charge of', 'look after']
      -[hold]),
      (['look after', 'keep under observation', 'keep an eye on']
      -[watch, 'look out']) ]).
```

`'Look after'` has three senses here, two of which can be extended to mean `'provide child care services'`. The third sense can be related to `'take care of'` through `watch`.

The rest of the sense shows that `'look after'` is similar in meaning to `bear`, `'care for'`, and `maintain`, which also appear in the text. In fact, `'look after'`, `'take care of'` and `'care for'` have been used interchangeably in the text, so they should be mapped to the same Case pattern when performing CtH conversion. Meanwhile however they should not be confused with `bear` and `maintain`, even though WordNet indicates they are likely to have the same sense.

4.5. Constructing equivalent classes of phrases

Although the similarity lists built in the previous phase give us a good deal of information to use when clustering words and phrases into classes of equivalent or similar meanings, the task is not as straightforward as one may expect. Intuitively, words or phrases that have the same intersection should be placed in the same cluster. However, this results in a too fragmented a clustering, as the following examples show:

```
similar([line, lodging, residence], % cluster of words or phrases
         noun,                            % type
        [abidance]).                   % shared senses
similar([lodging, residence],          % cluster of words or phrases
         noun,                            % type
        [abidance, residence, residency]).% shared senses
```

The first cluster differs from the second only in the addition of the word `line`, yet the two clusters have very different meanings. The sense of `abidance` in the first cluster means `compliance` and `conformation`, whereas in the second it means `'place of dwelling'`. The latter sense is the correct sense for the child care tax guide. However, given a different context or domain, the former may well be correct so it cannot be discounted completely. The next example shows a similar situation with verbs where the second cluster is appropriate for the chosen text.

```
similar([bear,continue,deem,have,'look after',maintain,support],
         verb,                            % POS
        [hold]).                       % shared senses
similar(['look after','take care of'],% cluster of words or phrases
         verb,                            % POS
        [watch]).                         % shared senses
```

Results from processing the 73 sentences in the Supplement to Child Care Expense Claims of the 1990 Canadian Tax Guide text indicate that only 42 of the 267 clusters are useful (Appendix B), a hit rate of about 16%[14]. This suggests that the knowledge available through a general purpose system like WordNet will not serve a narrow domain such as the computation of tax deduction well, and that additional knowledge must be brought to bear to link more of the clusters we have identified. We have some ideas about how to improve this performance. A larger text may produce more candidates for clustering. If we are unable to find direct 1-step links between words or phrases, we may find indirect *n*-step connections between them. Finally, we may be able to find connections between words or phrases by traversing other kinds of links in WordNet.

If we are unable to apply public domain sources more productively to technical texts, we will need to find another source for the knowledge we require. For want of an alternative this must come from the user, the ultimate repository of domain knowledge and commonsense reasoning. The design of a program to involve the reader in this effort appears in Figure 3.

The verb in each input Case Pattern could be compared with those in existing Case Patterns to see if they have related senses. A number of Case Patterns may qualify. If so, the user could be asked to select one with which to equate the input Case Pattern's verb. If there is only one the user's choice would be between acceptance or rejection. When a selection is made the association of the input and selected word or phrase together with their shared sense is placed in a `context` database, whose main function is to record these associations and to supply them as the default when the same words or phrases appear again.

The user might also perform other functions. For instance, the current design envisions a need to merge two related associations, e.g., create a larger cluster from `'take care of'`, `'look after'`, and `'care for'`. These phrases not currently contained in any single cluster. Functions which a user might wish to perform are these:

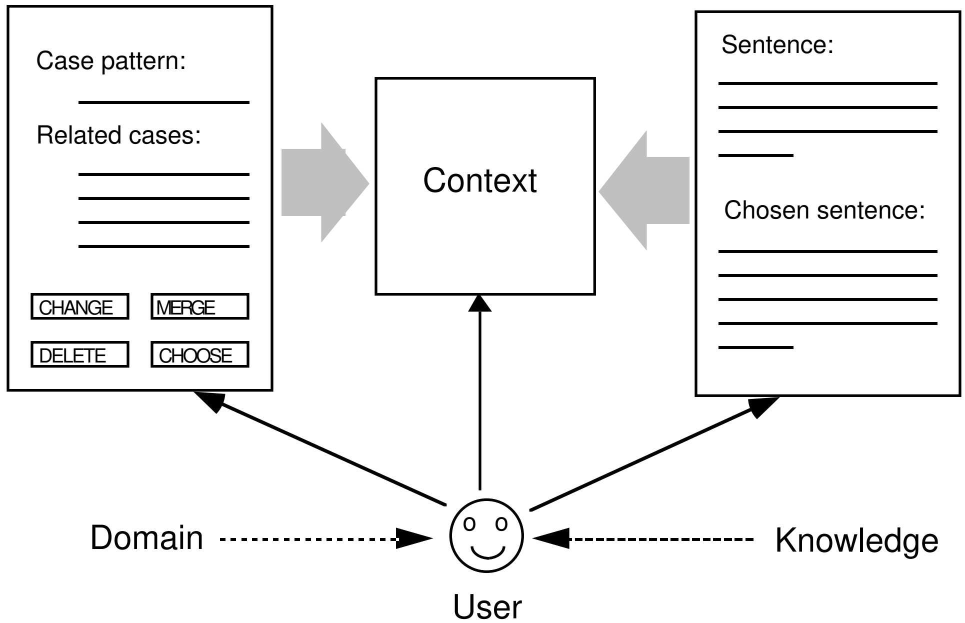

Figure 3. Involvement of the User in Word Sense Disambiguation

- CONFIRM. The user agrees to the default or a cluster by the system supplied automatically,
- REJECT. The user rejects any association between clusters,
- CHANGE. The user changes the axiomatization of the Case Patterns associated with the cluster,
- MERGE. The user merges several chosen clusters to make a larger one,
- DELETE. The user deletes a cluster and discards its associations,
- CHOOSE. The user selects one or more clusters.

5. Conclusions

This paper describes ongoing projects in knowledge extraction from text underway at the Computer Science Department of the University of Ottawa. It shows that the utilization of several public resources in Natural Language Understanding can help alleviate problems associated with interpreting various senses of natural language phrases and thus help produce good associations between Case pattern structures. Our guiding philosophy has been to involve the user as the ultimate source of knowledge. To that end we have designed tools to reduce the burden of interaction on him, e.g., by clustering similar words or phrases in word sense disambiguation. We hope that a consistent and user-friendly toolkit can perform nontrivial knowledge extraction from text when integrated with the overall, developing system.

6. Acknowledgment

The authors kindly acknowledge the support of the Natural Sciences and Engineering Research Council of Canada for this work through the Strategic Grant STR 0117764. The authors are grateful to the other members of the TANKA/MaLTe team, i.e., Ken Barker, Peter Clark, Jean-François Delannoy, Sylvain Delisle, Louis Massey, Ouerd Messaouda.

---

[1] We have not yet made up our minds on the nature of deep semantics, whether it is homogenous or in some way heterogenous.

[2] The 1991 Revenue Cnaada Income Tax guide. Specifically the sections dealing with credit for child care expenses.

[3] POS is an abbreviation for 'part of speech'.

[4] Sorting Collins reduced it from an advertised 'over 317000 English words' to 311,587.

[5] v1.4 is due for release soon (personal communication from Randee Tengi of the WordNet project, Aug. 7, 1993).

[6] known to be out of date.

[7] An alternative approach halfway between full interaction and full automation would be to perform dictionary lookup as a preprocess to parsing. This would avoid the delay of up to a couple of minutes between interactions which the user currently experiences while DIPETT is parsing the current input.

[8] Collins recognizes nine POS, DIPETT's dict/4 database distinguishes 25.

[9] phrases whose meanings that cannot be predicted fromthose of the words that compose them.

[10] but alas not in all.

[11] One reason is the difficulty in constructing *n*-ary predicates when the verb on which the predicate name is based occurs with different numbers of Cases. The sentences `'Jim lives with his family in 1980'` and `'Jim lives with his family'` produce patterns with 4 and 3 Cases respectively: `live-agt-acmp-tat` and `live-agt- acmp`.

[12] Words with of 'other ' POS are ignored in this process.

[13] The exact meaning of `'take care of'` in the child care tax guide is *'to provide child care services in exchange for payment'*.

[14] Unfortunately the child care tax guide did not contain enough word or phrase instances to properly test our other idea of disambiguating words by intersecting their sense lists and selecting the sense shared by most other word in the text.

References

Copeck T., S. Delisle & S. Szpakowicz (1992). “Parsing and Case Analysis in Tanka”, *Proceedings of the 15th Annual Conference on Computational Linguistics* (COLING-92), Nantes, July, pp. 1008-1012.

Delannoy J.F., C. Feng, S. Matwin & S. Szpakowicz (1993). “Knowledge Extraction from Text: Machine Learning for Text-to-rule Translation”, *Proceedings of the Workshop on Machine Learning Techniques and Text Analysis, European Conference on Machine Learning* (ECML-93), Vienna, April 8, pp. 1-7.

Delisle, S. & S. Szpakowicz (1991). “A Broad-Coverage Parser for Knowledge Acquisition from Technical Texts”. E. Johnson (ed.) *Proceedings of the Fifth International Conference on Symbolic and Logical Computing*, Dakota State University, pp. 169-183.

Delisle, S., K. Barker, T. Copeck, & S. Szpakowicz (1992). “Interactive Semantic Analysis in TANKA: Designing HAIKU”, TR-92-32, Department of Computer Science, University of Ottawa.

Delisle, S., T. Copeck, S. Szpakowicz & K. Barker (1993). "Interactive Semantic Analysis of Technical Texts: Case Pattern Acquisition", TR-93-13, Department of Computer Science, University of Ottawa.

Fromkin, V. & R. Rodman (1988). *An Introduction to Language, Fourth Edition.* Holt, Rinehart and Winston, Inc.

Karp, D., Y. Shabes, M. Zaidel & D. Egedi (1992). “A Freely Available Wide Coverage Morphological Analyzer for English”. *Proceedings of the 15th Annual Conference on Computational Linguistics* (COLING-92), Nantes, July, 950-955.

Matwin, S. & S. Szpakowicz (1993). “Text Analysis: How Can Machine Learning Help?”, *Proceedings of the First Conference of the Pacific Association for Computational Linguistics* (PACLING),Vancouver, April 21-24, pp. 33-42.

Matwin, S. & S. Szpakowicz (1992). "Machine Learning Techniques in Knowledge Acquisition from Text". *THINK*, 1(2), pp. 37-50.

Miller, G., R. Beckwith, C. Feilbaum, D. Gross & K.J. Miller (1990). “Introduction to WordNet: An On-line Lexical Database”. *International Journal of Lexicography*, 3(4), pp. 235-244.

Quirk R., S. Greenbaum, G. Leech & J. Svartvik (1985). *A Comprehensive Grammar of the English Language.* Longman, New York.

Appendix A. The Top Three Levels of the WordNet Noun Hyponymy

- thing, entity
  - act, human action, human activity
    - thing
    - accomplishment
    - nonaccomplishment
    - change
    - change of shape
    - rejection
    - activity, behavior
    - judgment, judgement
    - disapproval
    - production
    - stay
    - inactivation, inactivity
    - group action, interaction, social activity
  - abstraction
    - attribute
    - measure, quantity, amount, quantum
    - relation
    - space
    - time
  - entity
    - agent, cause, causal agency, causal agent
    - cell
    - life form, organism, being
    - object, inanimate object, physical object, thing
    - part, piece
    - unit, building block
    - whole, whole thing, unit
  - event, natural event
    - happening, occurrence
    - miracle
    - social event
    - trouble
  - group, grouping
    - Allies
    - array, arrangement
    - biological group
    - collection, pack, aggregation, accumulation, assemblage
    - kingdom
    - people, folk
    - system
    - world, human race, humanity, humankind, mankind
  - location
    - home
    - imaginary place
    - line
    - pass, mountain pass, saddleback
    - point
    - region, area
    - surface
  - phenomenon, natural phenomenon
    - act of God, force majeure, vis major, inevitable accident, unavoidable casualty
    - chemical phenomenon
    - consequence, effect, outcome, result, upshot
    - geological phenomenon
    - luck, fortune, chance, hazard
    - organic phenomenon
    - physical phenomenon
    - process
  - possession
    - ownership
    - property, belongings, holdings, material possessions
    - territory, dominion, territorial dominion, province, mandate, colony
    - white elephant
    - transferred property, transferred possession
    - asset
    - liability, financial obligation, indebtedness, pecuniary obligation
    - document
  - psychological feature
    - cognition, knowledge
    - motivation, motive, need
    - feeling, emotion
  - state
    - being, beingness, existence, existing
    - cognitive state, state of mind
    - condition
    - confinement
    - dead letter, non-issue
    - degree, stage, level, point
    - dependence, dependance, dependency
    - disorder, agitation
    - energy level, energy state
    - freedom, liberty
    - grace, state of grace
    - illumination
    - immaturity, immatureness
    - imperfection, imperfectness
    - independence, independency
    - maturity, matureness
    - motion
    - motionlessness, stillness
    - nonbeing, nonexistence
    - order
    - perfection, flawlessness, ne plus ultra
    - readiness, prepared
    - separation
    - situation, state of affairs
    - soundness
    - state of matter
    - status, social state
    - taboo, tabu
    - temporary state
    - union
    - unsoundness

Appendix B. Part of Semantic Clusters Found in the Child Care Tax Guide

```
similar( '[lodging,residence]', noun, [abidance,residence,residency]).
similar('[home,residence]', noun,
   [abode,address,domicile,dwelling,habitation,home,residence]).
similar('[individual,person]', noun,
   [agent,being,cause,human,individual,man,mortal,organism,person,someone,soul]).
similar('[assistance,service]', noun,[aid,assistance,help,helpfulness,helping]).
similar('[part,portion]', noun,
   [asset,dole,fraction,object,part,percentage,portion,share,thing]).
similar('[country,state]', noun,
   [body_politic,commonwealth,country,land,nation,republic,state]).
similar('[lodging,school]', noun,[building,edifice]).
similar([employment,job,work]', noun,[business,employment,job,line,occupation,work]).
similar('[daughter,son]', noun, [child]).
similar('[husband,spouse,wife]', noun, [consort,mate,partner,spouse]).
similar('[day_care_center,day-care_center]', noun,
   [day_care_center,day_nursery,nursery]).
similar('[cost,expense]', noun,
   [disbursal,disbursement,expenditure,expense,outgo,outlay,spending]).
similar('[portion,section]', noun, [division,fraction,part,portion]).
similar('[educational_institution,school]', noun,
   [educational_institution,establishment,institution,school]).
similar([force,service]', noun, [force,personnel]).
similar('[part,section]', noun, [fraction,part,portion]).
similar('[income,net_income]', noun, [income]).
similar('[brother,member]', noun, [member]).
similar('[benefit,payment]', noun, [pay,payment]).
similar('[force,work]', noun, [power]).
similar('[province,state]',noun, [province,state]).
similar('[home,lodging]', noun, [residence]).
similar('[educational_institution,nursery_school,school]', noun, [school]).
similar('[child_care,service]', noun, [service,services]).
similar('[income_tax,tax,taxation]', noun, [tax,taxation]).
similar('[be,live]', verb, [be,be_alive,exist,have_life,live]).
similar('[board,live_in]', verb, [board]).
similar('[care_for,look_after]', verb,[care_for,give_care_to,provide_care_for]).
similar('[live,live_in]', verb, [live_in]).
similar('[look_after,take_care_of]', verb, [watch]).
```