Comments and Code as Bitext

James Paul White
jimwhite@uw.edu
Department of Linguistics (CLMS)
University of Washington
A Long and Winding Road

Computer Language as Human Language

The Linguistic Interpretation of Computer Programs (and Related Text)

Jim White
February 3, 2012

A CLMS Thesis Proposal
Under Construction
Apply the Methods of Computational Linguistics to the Problems of Computer Program Understanding

Acquire Natural Language and World Knowledge by Reading Computer Programs and/or Related Text

A Brief Personal History and Some Philosophy

Choosing a Problem and an Approach

Computer Programs and Related Text
1993 a landmark year in empirical methods for processing parallel corpora

- Included C source code for the program *align*.

- IBM word alignment models 1 thru 5

First Workshop on Very Large Corpora
- [http://aclweb.org/anthology/sigdat.html#1993_0](http://aclweb.org/anthology/sigdat.html#1993_0)
- Led to first conference on Empirical Methods in Natural Language Processing in 1996
What is Bitext?


- Originally used to refer to documents and their translations into other languages for use in translation studies

- Now commonly used to refer to a broader range of parallel resources including multiple translations in same language

- Related Terms
  - Parallel Text: same as bitext in CL, comparable in TS
  - Comparable Text: same domain, differing languages
  - Parallel Corpora: collections of bitext
  - Translation Corpora: stricter form of parallel corpora
Bitext in the Wild

  - A “completely unsupervised” method for extracting parallel sentences from quasi-comparable corpora.
  - A noisy parallel corpus has documents which contain many parallel sentences in roughly the same order.
  - Comparable corpora contain topic-aligned documents which are not translations of each other.
  - Quasi-comparable corpora contain bilingual documents which are not necessarily on the same topic.

  - Wikipedia is a “surprisingly” useful source of parallel data.
Computer programs and related text are a vast untapped source of bitext

Billions of lines of Open Source Software and growing exponentially since 1995 (and until ?)

Collocated with the human language labels, strings, and comments in source files, programmers create elaborate semantic annotations in computer language

There are then document-level relationships between source files, data files, and documentation files

Bitext in which one of the languages is a computer language is very useful because there are many things a computer can do with language it can parse reliably and evaluate
As with other noisy parallel, comparable, and quasi-comparable corpora there are many and varied forms of translational relationships between the texts.

The Pedantic Javadoc Corpus is a small bitext corpus (currently 100 pairs) of English comments and simple Java statements.

Created to begin exploring the semantic representation issues and as a test for automatic classification of such pairs.

One application I envision would use SMT methods to generate code for trials within a reinforcement learning framework for following instructions such as README files.
Three Level Models for Conceptual Representation

Popular Models in Psychology/Cognitive Science/Artificial Intelligence

<table>
<thead>
<tr>
<th>Roger Schank &amp; Robert Abelson</th>
<th>David Marr</th>
<th>Zenon Pylyshyn</th>
<th>Glass, Holyoak, and Santa</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal</td>
<td>Computational</td>
<td>Semantic</td>
<td>Content</td>
</tr>
<tr>
<td>Plan</td>
<td>Algorithmic</td>
<td>Syntactic</td>
<td>Form</td>
</tr>
<tr>
<td>Action</td>
<td>Implementational</td>
<td>Physical</td>
<td>Medium</td>
</tr>
</tbody>
</table>


Java

```java
void bubblesort(int[] s) {
    ...
}
```

Goal/Computational
- Sort the given list of numbers.
- Upon return, the list must be in sorted order with the smallest element first.
- This method sorts a list of integers into increasing order using Bubblesort.

Plan/Algorithmic
- Scan the list from beginning to end, swapping pairs where the first is larger than the second. Repeat until no swaps are performed.

Action/Implementational
- Set the swapped flag. While swapped is true, reset the flag then scan for pairs needing a swap. To scan for pairs needing a swap, with an index starting from zero and incrementing until it is the length of the array minus one, compare the numbers at the index and index plus one. If the first number is larger than the second, exchange them and set the swapped flag to true.
Get the "<B>" string.

```java
com.sun.tools.doclets.formats.html.markup.HtmlWriter

public String getBold() {
    return "<B>";
}
```

Return, text passed, with Italics <I> and </I> tags, surrounding it.

```java
public String italicsText(String text) {
    return "<I>" + text + "</I>";
}
```
Start a set of fresh registers.

```
com.sun.tools.javac.jvm.Code
void newRegSegment() {
    nextreg = max_locals;
}
```

(Vars
 ((((IDENTIFIER 0)(NAME maxlocals))
   (((IDENTIFIER 1)(NAME nextreg))))))

(Tree (EXPRESSION_STATEMENT
  (Expression (ASSIGNMENT
    (Expression (IDENTIFIER 0))
    (Variable (IDENTIFIER 1))))))
Is c a printable ASCII character?

```java
com.sun.tools.apt.mirror.declaration.Constants.Formatter
static boolean isPrintableAscii(char c) {
    return c >= ' ' && c <= '~';
}
```

- The Javadoc is a translation of the programmer’s phrase for the method’s name and therefore literal or pedantic in that sense
  - See the Java Programmer’s Phrase Book slide here later
- The Javadoc’s relationship to the method body though is mostly at goal level
  - The top-level is actually pedantic since “is” maps to “return” here
Interesting Examples

Increase left margin by indentation width.

```java
com.sun.tools.javac.tree.Pretty
void indent() {
    lmargin = lmargin + width;
}
```

Decrease left margin by indentation width.

```java
com.sun.tools.javac.tree.Pretty
void undent() {
    lmargin = lmargin - width;
}
```
A Not So Simple Example

Return code byte at position pc as an unsigned int.
com.sun.tools.javac.jvm.Code
int get1(int pc)
{
    return code[pc] & 255;
}

(RETURN
    (Expression (AND
        (LeftOperand (ARRAY_ACCESS
            (Expression (IDENTIFIER (Name code)))
            (Index (IDENTIFIER (Name pc))))
        (RightOperand (INT_LITERAL (Value 255)))))))
Return code byte at position pc as an unsigned int.
(RETURN(Expression(AND
   (LeftOperand(ARRAY_ACCESS
      (Expression (IDENTIFIER (Name code)))
      (Index (IDENTIFIER (Name pc))))
   (RightOperand(INT_LITERAL(Value 255)))))))

(Vars (((IDENTIFIER 0)(Name code)) ((IDENTIFIER 1)(Name pc)))))
(Tree (RETURN(Expression(AND
   (LeftOperand (ARRAY_ACCESS
      (Expression (IDENTIFIER 0)))
      (Index (IDENTIFIER 1))))
   (RightOperand (INT_LITERAL(Value 255)))))))
**Composing a Goal**

*Return true iff float number is positive 0.*

```java
com.sun.tools.javac.jvm.Items.ImmediateItem
boolean isPosZero(x)
float x
{
    return x == 0.0F && 1.0F / x > 0.0F;
}
```

(Vars (((IDENTIFIER 0) (Name x)) ((IDENTIFIER 1) (Name x)))) (Tree (RETURN (Expression (CONDITIONAL_AND (LeftOperand (EQUAL_TO (LeftOperand (IDENTIFIER 0)) (RightOperand (FLOAT_LITERAL (Value 0.0)))))) (RightOperand (GREATER_THAN (LeftOperand (DIVIDE (LeftOperand (FLOAT_LITERAL (Value 1.0))) (RightOperand (IDENTIFIER 1)))) (RightOperand (FLOAT_LITERAL (Value 0.0))))))))))
Bind Them Variables

Return true iff float number is positive 0.

return x == 0.0F && 1.0F / x > 0.0F;

<table>
<thead>
<tr>
<th>S</th>
<th>VP</th>
<th>PP</th>
</tr>
</thead>
<tbody>
<tr>
<td>VP</td>
<td>PP</td>
<td>P</td>
</tr>
<tr>
<td>V</td>
<td>AP</td>
<td>if</td>
</tr>
<tr>
<td>V</td>
<td>ADJ</td>
<td>N</td>
</tr>
<tr>
<td>return</td>
<td>true</td>
<td>N</td>
</tr>
<tr>
<td>float</td>
<td>N</td>
<td>ADJ</td>
</tr>
<tr>
<td>number</td>
<td>N</td>
<td>ADJ</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TOP</th>
<th>h1</th>
<th>INDEX</th>
<th>e3</th>
</tr>
</thead>
<tbody>
<tr>
<td>pronoun_q:0:42</td>
<td>_return_v_1:0:6</td>
<td>subord:7:11</td>
<td>_true_a_of:7:11</td>
</tr>
<tr>
<td>LBL</td>
<td>h4</td>
<td>LBL</td>
<td>h9</td>
</tr>
<tr>
<td>ARG0</td>
<td>x6</td>
<td>ARG0</td>
<td>e3</td>
</tr>
<tr>
<td>RSTR</td>
<td>h5</td>
<td>ARG1</td>
<td>x6</td>
</tr>
<tr>
<td>BODY</td>
<td>h7</td>
<td>LBL</td>
<td>h10</td>
</tr>
<tr>
<td>ARG0</td>
<td>e18</td>
<td>ARG0</td>
<td>e13</td>
</tr>
<tr>
<td>ARG1</td>
<td>h19</td>
<td>ARG1</td>
<td>i15</td>
</tr>
<tr>
<td>ARG2</td>
<td>h20</td>
<td>ARG2</td>
<td>i17</td>
</tr>
<tr>
<td>_if_x_then:12:14</td>
<td>udef_q:15:27</td>
<td>compound:15:27</td>
<td>udef_q:15:20</td>
</tr>
<tr>
<td>LBL</td>
<td>h2</td>
<td>LBL</td>
<td>h21</td>
</tr>
<tr>
<td>ARG0</td>
<td>e18</td>
<td>ARG0</td>
<td>x23</td>
</tr>
<tr>
<td>ARG1</td>
<td>h19</td>
<td>ARG1</td>
<td>x23</td>
</tr>
<tr>
<td>ARG2</td>
<td>h20</td>
<td>ARG2</td>
<td>x26</td>
</tr>
<tr>
<td>_float_n:1:15:20</td>
<td>LBL</td>
<td>h25</td>
<td></td>
</tr>
<tr>
<td>ARG0</td>
<td>x23</td>
<td>ARG0</td>
<td>x34</td>
</tr>
<tr>
<td>ARG1</td>
<td>x23</td>
<td>RSTR</td>
<td>h36</td>
</tr>
<tr>
<td>ARG2</td>
<td>x34</td>
<td>BODY</td>
<td>h37</td>
</tr>
<tr>
<td>_number_n_of:21:27</td>
<td>LBL</td>
<td>h32</td>
<td></td>
</tr>
<tr>
<td>ARG0</td>
<td>e33</td>
<td>ARG0</td>
<td>x34</td>
</tr>
<tr>
<td>ARG1</td>
<td>x34</td>
<td>RSTR</td>
<td>h42</td>
</tr>
<tr>
<td>ARG2</td>
<td>x34</td>
<td>BODY</td>
<td>h43</td>
</tr>
<tr>
<td>_positive_n:1:31:39</td>
<td>LBL</td>
<td>h38</td>
<td></td>
</tr>
<tr>
<td>ARG0</td>
<td>x34</td>
<td>ARG1</td>
<td>i45</td>
</tr>
<tr>
<td>CARG</td>
<td>&quot;TOP&quot;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HCONS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>h1=qh2, h5=qh8, h12=qh9, h11=qh14, h19=qh10, h20=qh32, h22=qh25, h29=qh31, h36=qh38, h42=qh44</td>
<td></td>
</tr>
</tbody>
</table>
Related Work

- All sorts of Computer Science and Software Engineering Research
  - Static & Dynamic Code Analysis
  - Formal Verification & Proof Methods
  - Requirements and Specifications Traceability
  - Program Understanding & Program Comprehension Tools
    - Currently directed at human programmers and effectiveness is typically evaluated using ethnographic and CogSci experiments

- Open Ontology / Folksonomy Reasoning
  - WordNet, Freebase, etc
  - The Semantic Web
accept, action, add, check, clear, close, create, do, dump, end, equals, find, get, has, init, initialize, insert, is, new, next, parse, print, process, read, remove, reset, run, set, size, start, update, validate, visit

**Fig. 4.** The is-* branch of phrases

**Table 5.** Distribution of grammatical structures

<table>
<thead>
<tr>
<th>Structure</th>
<th>Instances</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>[verb]-[noun+]</td>
<td>422546</td>
<td>39.45%</td>
</tr>
<tr>
<td>[verb]</td>
<td>162050</td>
<td>15.13%</td>
</tr>
<tr>
<td>[verb]-[type]</td>
<td>78632</td>
<td>7.34%</td>
</tr>
<tr>
<td>[verb]-[adjective]-[noun+]</td>
<td>74277</td>
<td>6.93%</td>
</tr>
<tr>
<td>[verb]-[adjective]</td>
<td>28397</td>
<td>2.65%</td>
</tr>
<tr>
<td>[noun+]</td>
<td>26592</td>
<td>2.48%</td>
</tr>
<tr>
<td>[verb]-[noun+]-[type]</td>
<td>18118</td>
<td>1.69%</td>
</tr>
<tr>
<td>[adjective]-[noun+]</td>
<td>15907</td>
<td>1.48%</td>
</tr>
<tr>
<td>[noun+]-[verb]</td>
<td>14435</td>
<td>1.34%</td>
</tr>
<tr>
<td>[preposition]-[type]</td>
<td>13639</td>
<td>1.27%</td>
</tr>
</tbody>
</table>


Future Work

- **Linux Package Corpus**
  - Package metadata curated by package maintainers
  - Includes download link, package dependency expressions, single line and single paragraph description summaries, unpack, build, and install scripts
  - Currently RPM, easily extend to Debian and Nix

- **Click & It Goes**
  - Apply Reinforcement Learning to generate build and install scripts from the README and INSTALL files
  - Test by comparing whether scripts generated for packages with dependencies result in their dependents still working
Thank You!

James Paul White
jimwhite@uw.edu
Department of Linguistics (CLMS)
University of Washington