Predicting English Keywords from Java Bytecodes

Pablo Ariel Duboue, PhD

Les Laboratoires Foulab
Montreal, Quebec

Séminaires RALI-OLST, Université de Montréal

Outline

Introduction

About the Speaker

Bytecodes as Weak Semantics

Reverse Engineering

Details

Corpus Assembly

Main Pipeline

Results

Applications

Other Topics

GRIUM/RALI

Other Academic

Focus on Technology

Outline

Introduction

About the Speaker

Bytecodes as Weak Semantics

Reverse Engineering

Details

Corpus Assembly

Main Pipeline

Results

Applications

Other Topics

GRIUM/RALI

Other Academic

Focus on Technology

Before Montreal

- Columbia University
  - WSD in biology texts (GENIES)
  - Natural Language Generation in medical and intelligence domains (MAGIC, AQUAINT)
  - Thesis: “Indirect Supervised Learning of Strategic Generation Logic”, defended Jan. 2005.
    - Advisor: Kathy McKeown
    - Committee: Hirschberg/Jurafsky/Rambow/Jebara
- IBM Research Watson
  - AQUAINT: Question Answering (PIQuAnT)
  - Enterprise Search - Expert Search (TREC)
  - Connections between events (GALE)
  - Deep QA - Watson

In Montreal

I am passionate about improving society through language technology and split my time between teaching, doing research and contributing to free software projects

- Working with Prof. Nie at GRIUM
- Taught a graduate class in NLG in Argentina
- Contributed to Free Software projects, including some of my own
- Doing some consulting focusing on startups

Outline

Introduction

- About the Speaker
- Bytecodes as Weak Semantics
- Reverse Engineering

Details

- Corpus Assembly
- Main Pipeline
- Results
- Applications

Other Topics

- GRIUM/RALI
- Other Academic
- Focus on Technology

Semantics, Java Bytecodes, Javadocs

- Motivation: Machine Learning for Natural Language Generation
  - Finding good semantic representations “in the wild” is very rare
    - Level of detail of semantic representations vs. natural language
    - Similarities with binary code and code comments
  - Reverse Engineering practitioners could tolerate noisy text
    - As discussed in the INLG panel last summer

Java Bytecodes

- JVM is a stack machine
- The set of opcodes (~200) is small to simplify porting to new architectures.
- The opcodes fall into six categories:
  - Load/store (e.g. aaload, bastore)
  - Arithmetic/logic (e.g. iadd, fcmpg)
  - Type conversion (e.g. i2b, f2d)
  - Object construction and manipulation (new, putfield)
  - Operand stack manipulation (e.g. swap, dup2_x1)
  - Control flow (e.g. if_icmpgt,goto)
  - Method invocation and return (e.g. invokedynamic, lreturn)

LDC and CALL

- While bytecodes represent a reduced vocabulary, they can incorporate names of classes or methods and string constants

ldc pushes a constant onto the operand stack (number or string)

getfield instance and field name

getstatic classname and field name

invokedynamic invokes a dynamic method

Javadocs

- Javadocs are standardized Java comments
  - Include special mark-up in the form of '@' constructions
    - @param, @throws, @return among others
- In my work, I focus on the comments associated with each method
- Example:
  - Creates a CacheRandom instance with a given cache capacity. @param capacity The capacity of the cache.
  - Adjusts the relative offset where the match begins to an absolute value. Only used by AwkMatcher to adjust the offset for stream matches. @return The length of the match.

Outline

Introduction

- About the Speaker
- Bytecodes as Weak Semantics
- Reverse Engineering

Details

- Corpus Assembly
- Main Pipeline
- Results
- Applications

Other Topics

- GRIUM/RALI
- Other Academic
- Focus on Technology

What is Reverse Engineering

- From Wikipedia

  > *Reverse engineering is the process of discovering the technological principles of a device, object, or system through analysis of its structure, function, and operation. (...) The same techniques are subsequently being researched for application to legacy software systems (...)* ***to replace incorrect, incomplete, or otherwise unavailable documentation***.

- REcon: the premier reverse engineering conference, held yearly at Montreal

Reverse Engineering Example

```
private final int c(int) {
    0 aload_0
    1 getfield org.jpc.emulator.f.v
    4 invokeinterface org.jpc.support.j.e()
    9 aload_0
    10 getfield org.jpc.emulator.f.i
    13 invokevirtual org.jpc.emulator.motherboard.q.e()
    16 aload_0
    17 getfield org.jpc.emulator.f.j
    20 invokevirtual org.jpc.emulator.motherboard.q.e()
    23 iconst_0
    24 istore_2
    25 iload_1
    26 ifle 128
    29 aload_0
    30 getfield org.jpc.emulator.f.b
    33 invokevirtual org.jpc.emulator.processor.t.w()
```

Reverse Engineering Example

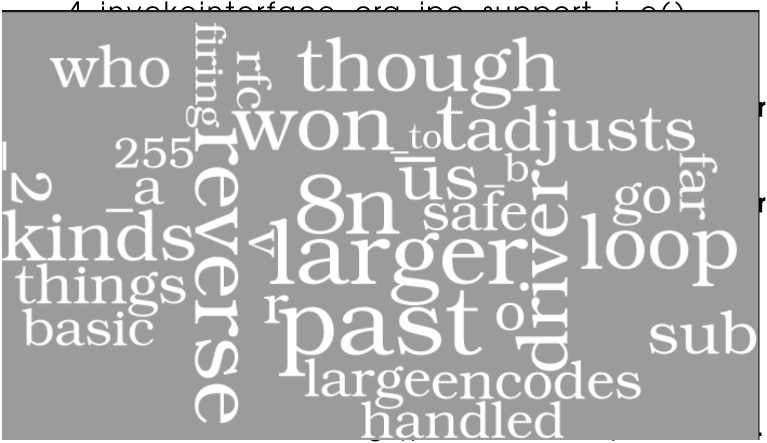
```
private final int c(int) {
     0 aload_0
     1 getfield org.jpc.emulator.f.v
     4 invokeinterface org.jpc.support.j.e()
     9 aload_0
     10 getfield org.jpc.emulator.f.i
     13 invokevirtual org.jpc.emulator.motherboard.q.e()
     16 aload_0
     17 getfield org.jpc.emulator.f.j
     20 invokevirtual org.jpc.emulator.motherboard.q.e()
     23 iconst_0
     24 istore_2
     25 iload_1
     26 ifle 128
     29 aload_0
     30 getfield org.jpc.emulator.f.b
     33 invokevirtual org.jpc.emulator.processor.t.w()
```

Reverse Engineering Example

```
private final int c(int) {
     0 aload_0
     1 getfield org.jpc.emulator.f.v
     4 invokeinterface org.jpc.support.j.e()
                                                     rd.q.e()

                                                     rd.q.e()

                                                     .t.w()
```



Reverse Engineering Example

```
private final int c(int) {
      0 aload_0
      1 getfield org.jpc.emulator.f.v
      4 invokeinterface org.jpc.support.j.e()
      9 aload_0
      10 getfield org.jpc.emulator.f.i
      13 invokevirtual org.jpc.emulator.motherboard.q.e()
      16 aload_0
      17 getfield org.jpc.emulator.f.j
      20 invokevirtual org.jpc.emulator.motherboard.q.e()
      23 iconst_0
      24 istore_2
      25 iload_1
      26 ifle 128
      29 aload_0
      30 getfield org.jpc.emulator.f.b
      33 invokevirtual org.jpc.emulator.processor.t.w()
```

Reverse Engineering Example

```
private final int c(int) {
     0 aload_0
     1 getfield org.jpc.emulator.f.v
     4 invokeinterface org.jpc.support.j.e()
                                                                  rd.q.e()
                                                                  rd.q.e()
                                                                  .t.w()
```

Outline

Introduction
- About the Speaker
- Bytecodes as Weak Semantics
- Reverse Engineering

Details
- Corpus Assembly
- Main Pipeline
- Results
- Applications

Other Topics
- GRIUM/RALI
- Other Academic
- Focus on Technology

Debian

- Using the Debian archive
  - `apt-file search --package-only .jar`
    - 1,400+ packages
  - `dpkg-query -p` *package name*
    - Look for `Source` field
  - `dpkg-source -x` *source .dsc*
    - Search for Java source files.
  - `dpkg -x` *binary .deb*
    - Search for jars, disassemble the methods.
- Assembling the Bytecodes / Javadoc Corpus
  - Disassemble using `jclassinfo --disasm`
  - Dump Javadoc comments using qdox.
    - A lightweight Java source parsing library.
  - Heuristically match source methods to compiled methods.
    - Normalize source code signatures to binary signatures.

Numbers

- Final corpus:
  - 1M methods.
  - 35M words.
  - 24M JVM instructions.
- This corpus is 3x bigger than the one discussed in the REcon talk

Outline

Introduction
- About the Speaker
- Bytecodes as Weak Semantics
- Reverse Engineering

Details
- Corpus Assembly
- Main Pipeline
- Results
- Applications

Other Topics
- GRIUM/RALI
- Other Academic
- Focus on Technology

Pipeline

1. HTML detagging
2. PTB tokenizer
3. Morfessor
4. cclparser
5. Naive Bayes

Morfessor

- Unsupervised morphome detection
  - http://www.cis.hut.fi/projects/morpho/
- CacheRandom → Cache + Random
- GenericCache.DEFAULT_CAPACITY → Generic + Cache. + DEFAULT_CAPACITY
- someFileName → some + FileName
- PatternStreamInput → Pattern + Stream + Input

CCL Parser

- CCL Parser is an unsupervised parser that does not require POS tags
  - Unsupervised POS induction, incremental (can deal with long sentences)
  - Yoav Seginer (2007), Fast Unsupervised Incremental Parsing. ACL.
  - http://www.seggu.net/ccl/
  - GPLv2 – but current codebase does not save trained models
- (((((((((((creates a) cache random) instance (with a)) given) cache) capacity. (@ param)) capacity the) capacity) of the) cache))
- As chunker
  - (creates a) (cache random) (instance) (with a) (given) (cache) (capacity.) (@ param) (capacity the) (capacity) (of the) (cache) (same) (as cache random) (generic) (cache.) (default_capacity)

Naive Bayes

- P(term | bytecodes)
- In case of complex opcodes (e.g., ldc “This is a very long string”), the count for the opcode is split between:
  - 0.5 for the full opcode, as a whole
  - 0.5 / #parts for each subpart ({ldc, This, is, a, very, long, string})

Outline

- Introduction
  - About the Speaker
  - Bytecodes as Weak Semantics
  - Reverse Engineering
- Details
  - Corpus Assembly
  - Main Pipeline
  - Results
  - Applications
- Other Topics
  - GRIUM/RALI
  - Other Academic
  - Focus on Technology

Overall Results

- Top scoring terms, using one count per opcode

Term	P	R	**F**
@ param	0.73	0.64	**0.685**
	0.73	*0.63*	***0.679***
object	0.97	0.06	**0.114**
@ throws	0.72	0.05	**0.099**
text	0.64	0.02	**0.038**
property	0.69	0.01	**0.031**
description	0.72	0.01	**0.029**
@ return the	0.78	0.01	**0.028**
	0.80	*0.01*	***0.026***

Without Per-opcode Normalization

Term	P	R	**F**
@ generated	0.76	0.80	**0.783**
replaced	0.93	0.60	**0.734**
@ param	0.64	0.74	**0.690**
icu	0.75	0.49	**0.600**
o the	0.47	0.75	**0.582**
@ stable	0.72	0.45	**0.561**
@ inheritdoc	0.42	0.60	**0.495**
@ return the	0.41	0.52	**0.463**
receiver	0.72	0.31	**0.440**

Where to go from here

- The meaning in the bytecodes is not in the presence of individual opcodes but in their sequencing
  - MOTIF analysis in bioinformatics
- Comparable SMT
  - Most systems (e.g., Munteanu and Marcu (2006)) use either an aligned corpora or a bilingual dictionary
  - I can try to obtain that by asking developers to write descriptions for segments of the code
- Alternatively, I can try to adapt TextTiling to bytecodes
  - Suggested by another Foulaber (Danukeru)

Outline

- Introduction
  - About the Speaker
  - Bytecodes as Weak Semantics
  - Reverse Engineering
- Details
  - Corpus Assembly
  - Main Pipeline
  - Results
  - Applications
- Other Topics
  - GRIUM/RALI
  - Other Academic
  - Focus on Technology

Applications in Reverse Engineering

- Hinting Subroutines
  - The motivating example at the beginning.
  - “Beacon identification” in Software Engineering.
- Custom (malware) VMs
  - Identifying which methods correspond to different VM operations (addition, jump, etc).
- Dalvik Word Clouds.
  - Use dex2jar, obtain word clouds for the whole executable.
  - Maybe the user can tell if anything looks fishy there?
- Flagging Suspicious Methods.
  - Finding methods that can be described with keywords very different from the rest of the existing methods.
  - Can be done with dynamically generated bytecodes.

Applications Outside Reverse Engineering

- Semantic Search
  - Searching for methods related to certain English terms
  - Query expansion using bytecodes
- Software Engineer Documentation
  - Generating documentation from bytecodes
  - Long term goal

Outline

Introduction
- About the Speaker
- Bytecodes as Weak Semantics
- Reverse Engineering

Details
- Corpus Assembly
- Main Pipeline
- Results
- Applications

Other Topics
- GRIUM/RALI
- Other Academic
- Focus on Technology

Snippets and Sentence Compression

- Improving Information Retrieval user experience and engine performance by having better **snippets**
  - Working closely with Dr. Jing He.
- Summarization snippets seem better than regular snippets but are much longer ⇒sentence compression
  - Query: wine rome
  - Page: `http://penelope.uchicago.edu/%7Egrout/encyclopaedia_romana/wine/wine.html`
    - Bing snippet: Return to Notae. Wine and Rome. Now nearly extinct in the wild, grapes (vitis vinifera) grew throughout the ancient Mediterranean, the juice readily fermenting as the enzymes ...
    - Summarization: Wine almost always was mixed with water for drinking; undiluted wine merum was considered the habit of provincials and barbarians. The earliest work on wine and agriculture was written in Punic. Indeed, by 154 BC, says Pliny, wine production in Italy was

Outline

Introduction

- About the Speaker
- Bytecodes as Weak Semantics
- Reverse Engineering

Details

- Corpus Assembly
- Main Pipeline
- Results
- Applications

Other Topics

- GRIUM/RALI
- Other Academic
- Focus on Technology

Taught Graduate Class in Argentina

- My alma mater
  - Universidad Nacional de Cordoba
- Natural Language Generation
  - http://wiki.duboue.net/index.php/2011_FaMAF_Intro_to_NLG
  - Touched NLG from DBs, Summarization and decoding in SMT
  - 12 students, about a fourth of the total PhD students in the dept
- Large NLP Group
  - http://pln.famaf.unc.edu.ar/
  - Possibilities for visiting people from Montreal

Student Projects

- Natural Language Generation for Software Patches
  - http://nlg4patch.com.ar/
- Natural Language Generation for UML diagrams
  - ongoing
- Referring Expression Evaluation using DBpedia
  - HLT-NAACL 2012 Short Paper “On The Feasibility of Open Domain Referring Expression Generation Using Large Scale Folksonomies”
- Surface Realization of Spanish using the SemPar Corpus

Outline

Introduction

- About the Speaker
- Bytecodes as Weak Semantics
- Reverse Engineering

Details

- Corpus Assembly
- Main Pipeline
- Results
- Applications

Other Topics

- GRIUM/RALI
- Other Academic
- Focus on Technology

Free Software

- Debian science
  - apertium, transfer-based machine translation for related language-pairs
- NLTK
- Personal Projects
  - Farmer text support
  - php-nlgen
  - NLG in Puredata
- http://www.ohloh.net/accounts/DrDub

Tech Scene Montreal

- Foulab
  - Montreal oldest and more prestigious hackerspace
    - http://foulab.org
    - Hackerspaces are community-operated physical places, where people can meet and work on their projects.
  - http://hackerspaces.org for the full list
  - Open House every Tuesday night, everybody is welcomed
- Hack-a-thons
  - Upcoming: http://quebecouvert.org/events/hackonslacorruption/
- Notman house
  - The “House of the Web” in Montreal
  - http://notman.org/

Consulting

- R&D for start-ups
  - Focusing on companies with positive contributions
  - Quick turnaround from ideas to users
  - http://honeypot.matchfwd.com
- Own ventures
  - 4opiniones.com

Summary

- I have presented a work-in-progress targetting the automated documentation generation from compiled code
  - Most recent progress is in unsupervised terminology identification
  - Currently working in improved ML

Acknowledgements

- GRIUM
  - Prof. Nie and Dr. Jing He
- Foulab
  - Danukeru
- REcon organizers
  - Subgraph.
- Annie Ying

Contacting the Speaker

- Email: pablo.duboue@gmail.com
- Website: http://duboue.net
- Twitter: @pabloduboue
- LinkedIn: http://linkedin.com/in/pabloduboue
- IRC: DrDub

http://keywords4bytecodes.org

- Always looking for new collaboration opportunities
  - Very interested in teaching a class either in Montreal or on-line