

BART: Baltimore Anaphora Resolution Toolkit

Emily K. Jamison

Department of Linguistics
The Ohio State University
jamison@ling.osu.edu

Slate Lab Meeting
Winter 2008



Outline

- 1 BART Summary
 - Introduction to Coreference Resolution
 - BART's modularity
 - MMAX2
- 2 How to Use BART
- 3 BART functionality, from Emily's perspective

Introduction to Coreference Resolution

What is Coreference?

Example

Queen Elizabeth set about transforming her husband, King George VI, into a viable monarch. Lionel Logue, a renowned speech therapist, was summoned to help the King overcome his speech impediment. (Ng 2003)

Coreference: 3 equivalence classes: QE, KG, LL

- QE: Queen Elizabeth, her
- KG: husband, King George VI, the King, his
- LL: Lionel Logue, a renowned speech therapist

Coreference Resolution: the problem of determining which noun phrases (NPs) refer to each real-world entity mentioned in a document.

Uses for Coreference Resolution

Why study Coreference Resolution?

- Question Answering
- Text Summarization
- Cross-Document Entity Coreference
- Information Extraction
- Machine Translation



Other Coreference Resolution Systems

Two other publicly-available systems:

JavaRAP (Qiu et al, 2004)

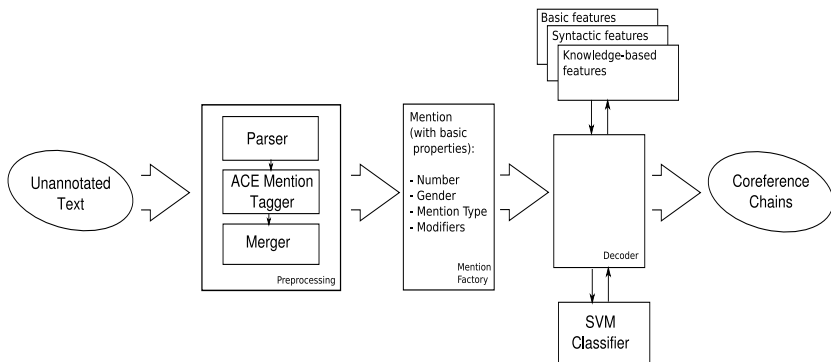
- Only resolves pronouns.
- Implementation of a single algorithm, RAP (Lappin and Leass 1994)
- <http://www-appn.comp.nus.edu.sg/rpnlpir/cgi-bin/JavaRAP/JavaRAPdemo.html>

GUITAR (Steinberger et al, 2007)

- Partial predecessor to BART
- Resolves a range of NP-type mentions
- includes ready-made algorithms such as a system for definite descriptions (Poesio and Vieira 2000), a system for proper names (Bontcheva et al 2002), and MARS for pronouns (Mitkov 1998).

GENERIC-RESOLVE

BART: a modular system



Example system configuration

Modularity: Preprocessing Options

Several parsing options currently available with BART:

- Charniak and Johnson's Reranking Parser (Charniak and Johnson 2005)
- Stanford Lexicalized Parser (Klein and Manning 2003)
- Berkeley Parser (Petrov et al 2006)

Or, use chunking:

- Stanford POS tagger + YamCha Chunker (Toutanova et al 2003) and (Kudoh and Matsumoto 2000)



Modularity: Feature Detectors

Lots of feature detectors available with BART:
Basic Features (Soon et al 2001):

- MentionType and MentionTypeBuggy
- Gender
- Number
- Alias
- Appositive
- String Matching
- SemanticClass
- SentenceDistance / DistDiscrete

Modularity: Feature Detectors

And More Features!

Syntactic Features:

- SynPos: Syntactic Position
- TreeFeature

Knowledge-based Features:

- WebPatterns
- WikiAlias: Wikipedia redirects for name variation
- Wiki: Wikipedia redirects and category graph to judge relatedness
- WikInc: Wikipedia and Wordnet to compute modifier compatibility
- WNSimilarity
- SemClassValue

Modularity: Learning Algorithms

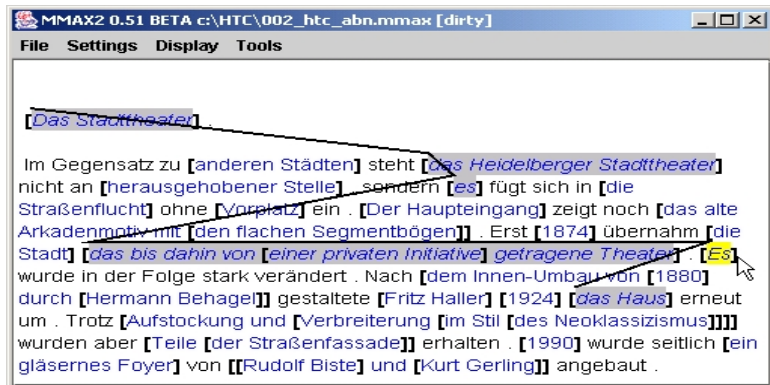
BART includes range of machine learning possibilities:

- WEKA machine learning toolkit. All classifiers are usable.
- SVMLight/ SVMLightTK (which handles tree-valued features)
- a maximum entropy classifier



MMAX2

MMAX2: an annotation tool from EML Research



Using MMAX2: Point and Click

Setting up BART

BART is installed in Linguistics /home/compling/BART

Note: Preprocessing currently doesn't work; needs YamCha chunker. Training and testing work.

IF you decide you want an installation elsewhere:

- read the BART "Description" in
/home/compling/BART/README1.txt (Coming Soon!)

Then...

- read Emily's additional installation instructions in
/home/compling/BART/README2.txt (Coming Soon!)

Note: BART will not work unless you follow the additional instructions in README2. This is why BART hasn't been released yet.



Setting up BART

Once BART is set up, it is simply run from the command line:

- `bash$ java -Xmx1024M elkfed.main.XMLExperiment`

Or,

- `bash$ java -Xmx1024M elkfed.main.XMLTrainer`
- `bash$ java -Xmx1024M elkfed.main.XMLClassifierBuilder`
- `bash$ java -Xmx1024M elkfed.main.XMLAnnotator`

Results of the testing phase are printed to stdout (after a good deal of other stuff).



Changing BART options with the XML files

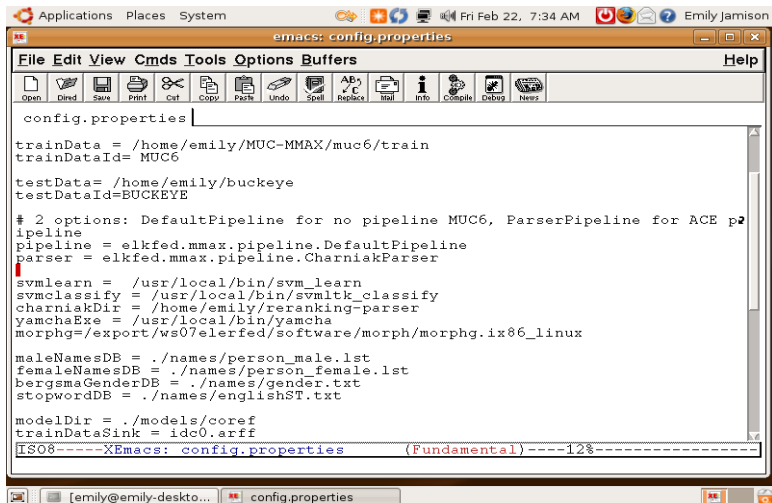
The config.properties file:

- training and test set locations
- preprocessing pipelines
- feature sets
- machine learning options



Changing BART options with the XML files

The config.properties file:



```
config.properties

trainData = /home/emily/MUC-MMAX/muc6/train
trainDataId= MUC6

testData= /home/emily/buckeye
testDataId=BUCKEYE

# 2 options: DefaultPipeline for no pipeline MUC6, ParserPipeline for ACE pipeline
pipeline = elkfed.mmax.pipeline.DefaultPipeline
parser = elkfed.mmax.pipeline.CharniakParser

svmlearn = /usr/local/bin/svm_learn
svmclassify = /usr/local/bin/svmltk_classify
charniakDir = /home/emily/reranking-parser
yamchaExe = /usr/local/bin/yamcha
morphg=/export/ws07elerfed/software/morph/morphg.ix86_linux

maleNamesDB = ./names/person_male.lst
femaleNamesDB = ./names/person_female.lst
bergsmaGenderDB = ./names/gender.txt
stopwordDB = ./names/englishST.txt

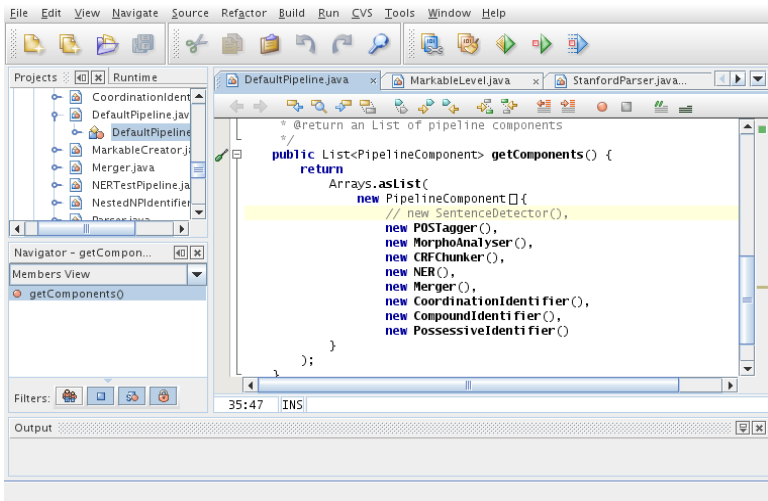
modelDir = ./models/coref
trainDataSink = idc0.arff

ISO8-----XEmacs: config.properties (Fundamental)-----12%-----
```

Changing individual features or preprocessing options

To change individual features or preprocessing options, you need to comment the code. But, features and options are clearly labelled, once you find the right file.

Changing individual features or preprocessing options



The screenshot shows an IDE window with the following components:

- Menu Bar:** File, Edit, View, Navigate, Source, Refactor, Build, Run, CVS, Tools, Window, Help
- Toolbar:** Standard IDE icons for file operations and development.
- Projects Panel:** Shows a tree view of project files including CoordinationIdentifier.java, DefaultPipeline.java, MarkableCreator.java, Merger.java, NERTestPipeline.java, and NestedNPIdentifier.java.
- Navigator - getCompon...:** Shows the current file and its members, with getComponents() selected.
- Members View:** Shows the selected member, getComponents().
- Code Editor:** Displays the following Java code:

```
    * @return an List of pipeline components
    */
    public List<PipelineComponent> getComponents() {
        return
            Arrays.asList(
                new PipelineComponent[] {
                    // new SentenceDetector(),
                    new POSTagger(),
                    new MorphoAnalyser(),
                    new CRFChunker(),
                    new NER(),
                    new Merger(),
                    new CoordinationIdentifier(),
                    new CompoundIdentifier(),
                    new PossessiveIdentifier()
                }
            );
    }
```
- Status Bar:** Shows the cursor position as 35:47 INS.
- Output Panel:** Empty.

What Emily has been able to do with BART

BART: an enormous engineering effort

- High functionality; high research potential
- High learning/ set-up curve



What Emily has been able to do with BART

MMAX2:

- helpful Quick Start Guide (Muller 2005)
- however, not enough specification in the guide to start a project from scratch. I had to use existing projects as templates. Hopefully, you have an existing project you can use as a template.
- Tip: Click on the antecedent, THEN the anaphor to mark coreference. Otherwise, you destroy the antecedent set.



What Emily has been able to do with BART

BART: Preprocessing Pipeline

- Chunking system works, with standard written text (i.e., punctuation, etc.)
- I can't get chunking system to work reliably with non-standard text
- Current release needs a patch for the parsing system to work



What Emily has been able to do with BART

BART: Training and Testing

- Reliably usable.
- Success with a range of learners, from MaxEnt to WEKA Decision Trees to a perceptron I wrote myself
- Results are printed to stdout for easy observation

However:

- Fancier features are deceptively hard to use (Ex: WikiAlias needs access to a MySQL database that contains redirect and link info. Not included with BART!)
- Time. WebSearch feature can take 7 seconds per anaphor/antecedent pair, which means over one day per typical 3 page file; over 33 days for the BART's MUC test sample...

Outline

- 1 BART Summary
 - Introduction to Coreference Resolution
 - BART's modularity
 - MMAX2
- 2 How to Use BART
- 3 BART functionality, from Emily's perspective