Apache UIMA and the Watson Jeopardy!$^{\text{TM}}$ System

Big Data Montreal Meetup

Pablo Ariel Duboue

Les Laboratoires Foulab
999 Rue du College
Montreal, H4C 2S3, Quebec

February 4th 2014

Outline

Watson
- Jeopardy!^{TM}
- Approach

Apache Unstructured Information Management Architecture
- Advantages
- Mini-Tutorial
- UIMA Asynchronous Scale-out (Low-latency)

My Own Personal Contributions
- To Watson
- After Watson

Outline

Watson

Jeopardy!^{TM}

Approach

Apache Unstructured Information Management Architecture

Advantages

Mini-Tutorial

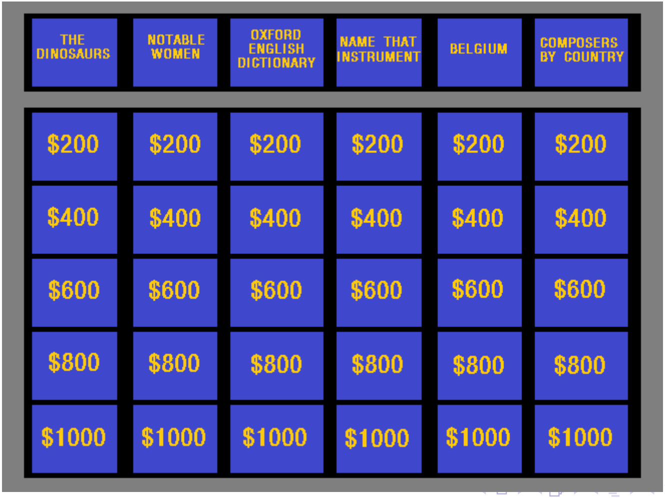
UIMA Asynchronous Scale-out (Low-latency)

My Own Personal Contributions

To Watson

After Watson

Problem



THE DINOSAURS	NOTABLE WOMEN	OXFORD ENGLISH DICTIONARY	NAME THAT INSTRUMENT	BELGIUM	COMPOSERS BY COUNTRY
$200	$200	$200	$200	$200	$200
$400	$400	$400	$400	$400	$400
$600	$600	$600	$600	$600	$600
$800	$800	$800	$800	$800	$800
$1000	$1000	$1000	$1000	$1000	$1000

Example Questions

Category: "J.P."
He played Duke Washburn, Curly's twin brother, in "City Slickers II".

- Answer: Jack Palance

About the Speaker

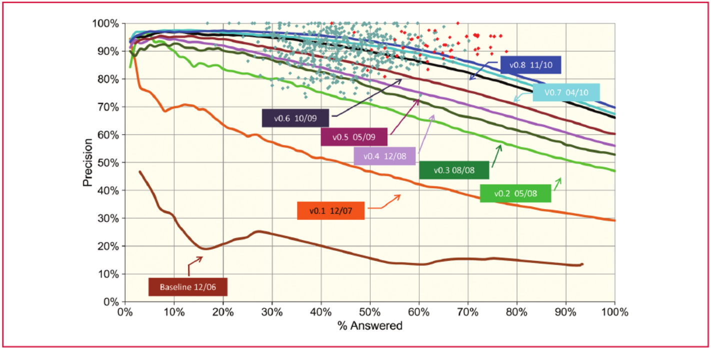
I am passionate about improving society through language technology and split my time between teaching, doing research and contributing to free software projects

- Columbia University
  - Natural Language Generation
  - Thesis: “Indirect Supervised Learning of Strategic Generation Logic”, defended Jan. 2005.
- IBM Research Watson
  - Question Answering
  - Deep QA - Watson
- Independent Research, living in Montreal
  - Collaboration with Universite de Montreal
  - Free Software projects and consulting for small companies

The Challenges of a Research Team

- Blazzingly fast development
  - Quick experimental turn-around is not a “nice to have” feature, it is key
- Dead code
- Lack of documentation
- Reproducibility of results

Architecture



Incremental progress from June 2007 to November 2010, from Ferrucci (2012)

The Challenges of a Grand Challenge

- Very expensive.
- Constantly on the verge of being canceled.
- Plenty of issues beyond the control of the research team.

Outline

Watson

- Jeopardy!^{TM}
- Approach

Apache Unstructured Information Management Architecture

- Advantages
- Mini-Tutorial
- UIMA Asynchronous Scale-out (Low-latency)

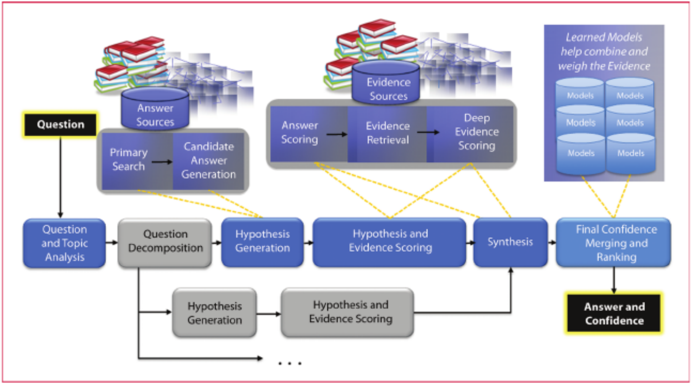
My Own Personal Contributions

- To Watson
- After Watson

Approach

- Keep all options open till the end
  - Do not overcommit
- Propose candidate answers by perfoming searches
- Gather supporting evindence by performing a search for each candidate answer (!)
- Analyze all this wealth of information in parallel
- Central scoring and ranking using Machine Learning

Architecture



DeepQA Architecture, from Ferrucci (2012)

Components Descriptions

Question Analysis. Extract keywords, assign to known classes, expand entities.

Primary Search. Obtain a set of documents relevant to the question.

Candidate Answer Generation. Extract from the documents candidate answers.

Evidence Retrieval and Scoring. Fetch passages (sentences) containing the candidate answers and relevant keywords, then score the candidates in context.

Final Confidence Merging. Apply a trained model based on the evidence.

Outline

Watson

- Jeopardy!$^{\text{TM}}$
- Approach

Apache Unstructured Information Management Architecture

- **Advantages**
- Mini-Tutorial
- UIMA Asynchronous Scale-out (Low-latency)

My Own Personal Contributions

- To Watson
- After Watson

Frameworks

- Frameworks enable:
  - Sharing and Collaboration
  - Growth
  - Deployment and Large scale implementations
  - Adoption
- Frameworks need:
  - Maintenance (no software is ever “completed”)
  - Documentation (to further collaboration / adoption)
  - Neutrality (w.r.t. applications being implemented)
  - Ownership (on behalf of their developers / maintainers)
  - Publicity (for widespread adoption)

Enabling Sharing and Collaboration

- Sharing within an organization
  - Code is the documentation
  - Agile sharing
  - Convention-over-configuration
- Sharing with the world
  - Enabling the greater good, without paying a high price (support time, spoiling potential ventures)
- Sharing with new / potential partners
  - Bringing new people up to speed
  - Attracting talent

Enabling Growth

- New phenomena
  - From syntactic parsing to semantic parsing
  - From parsing sentences to parsing USB traffic data
- New artifacts
  - From text to speech
- New architectures
  - From Understanding to Generation

Enable Deployment and Large scale implementations

- Multiple architectures
    - Windows, Linux
- On-line vs. off-line
    - Batch corpus processing vs. user-oriented Web services
- New programming languages (and old, efficient ones)
- New human languages

What is UIMA

- UIMA is a framework, a means to integrate text or other unstructured information analytics.
- Reference implementations available for Java, C++ and others.
- An Open Source project under the umbrella of the Apache Foundation.

Analytics Frameworks

- Find all telephone numbers in running text

  -
    ```
    (((\([0-9]{3}\))|[0-9]{3})-?
    [0-9]{3}-?[0-9]{4}
    ```
- Nice but...
  - How are you going to feed this result for further processing?
  - What about finding non-standard proper names in text?
  - Acquiring technology from external vendors, free software projects, etc?

In-line Annotations

- Modify text to include annotations
  - This/DET happy/ADJ puppy/N
- It gets very messy very quickly
  - (S (NP (This/DET happy/ADJ puppy/N) (VP eats/V (NP (the/DET bone/N)))
- Annotations can easily cross boundaries of other annotations
  - He said **<confidential>**the project can't go on. The funding is lacking.**</confidential>**

Standoff Annotations

- Standoff annotations
  - Do not modify the text
  - Keep the annotations as offsets within the original text
- Most analytics frameworks support standoff annotations.
- UIMA is built with standoff annotations at its core.
- Example:

```
He said the project can’t go on.  The funding is lacking.
_________________________________________________________

012345678901234567890123567890123456789012345678901234567
```

  - Sentence Annotation: 0-32, 35-57.
  - Confidential Annotation: 8-57.

Type Systems

- Key to integrating analytic packages developed by independent vendors.
- Clear metadata about
  - Expected Inputs
    - Tokens, sentences, proper names, etc
  - Produced Outputs
    - Parse trees, opinions, etc
- The framework creates an unified typesystem for a given set of annotators being run.

UIMA Advantages

- CAS
  - Memory Efficiency
  - Indices
- Types
- Interoperability
- Lean protocol serialization
  - UIMA AS sends and retrieves from network nodes only the required information
  - (default XMI serialization is anything but lean)

Outline

- Watson
  - Jeopardy!^{TM}
  - Approach
- Apache Unstructured Information Management Architecture
  - Advantages
  - Mini-Tutorial
  - UIMA Asynchronous Scale-out (Low-latency)
- My Own Personal Contributions
  - To Watson
  - After Watson

UIMA Concepts

- Common Annotation Structure or CAS
  - Subject of Analysis (SofA or View)
  - JCas
- Feature Structures
  - Annotations
- Indices and Iterators
- Analysis Engines (AEs)
  - AEs descriptors

Room annotator

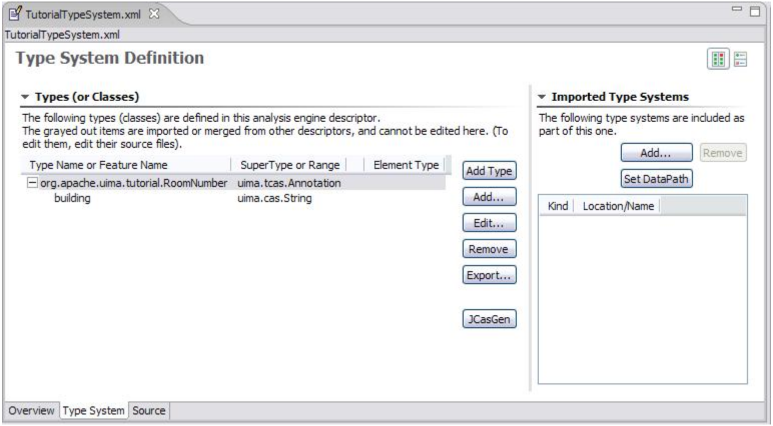
- From the UIMA tutorial, write an Analysis Engine that identifies room numbers in text.

  Yorktown patterns: 20-001, 31-206, 04-123 (Regular Expression Pattern: [0-9][0-9]-[0-2][0-9][0-9])

  Hawthorne patterns: GN-K35, 1S-L07, 4N-B21 (Regular Expression Pattern: [G1-4][NS]-[A-Z][0-9])

- Steps:
  1. Define the CAS types that the annotator will use.
  2. Generate the Java classes for these types.
  3. Write the actual annotator Java code.
  4. Create the Analysis Engine descriptor.
  5. Test the annotator.

Editing a Type System



TutorialTypeSystem.xml

Type System Definition

Types (or Classes)

The following types (classes) are defined in this analysis engine descriptor.
The grayed out items are imported or merged from other descriptors, and cannot be edited here. (To edit them, edit their source files).

Type Name or Feature Name	SuperType or Range	Element Type
org.apache.uima.tutorial.RoomNumber	uima.tcas.Annotation	
building	uima.cas.String	

Imported Type Systems

The following type systems are included as part of this one.

Kind	Location/Name

The XML descriptor

```
<?xml version="1.0" encoding="UTF–8" ?>
  <typeSystemDescription xmlns="http://uima.apache.org/resourceSpecifier">
    <name>TutorialTypeSystem</name>
    <description>Type System Definition for the tutorial examples –
        as of Exercise 1</description>
    <vendor>Apache Software Foundation</vendor>
    <version>1.0</version>
    <types>
      <typeDescription>
        <name>org.apache.uima.tutorial.RoomNumber</name>
        <description></description>
        <supertypeName>uima.tcas.Annotation</supertypeName>
        <features>
          <featureDescription>
            <name>building</name>
            <description>Building containing this room</description>
            <rangeTypeName>uima.cas.String</rangeTypeName>
          </featureDescription>
        </features>
      </typeDescription>
    </types>
  </typeSystemDescription>
```

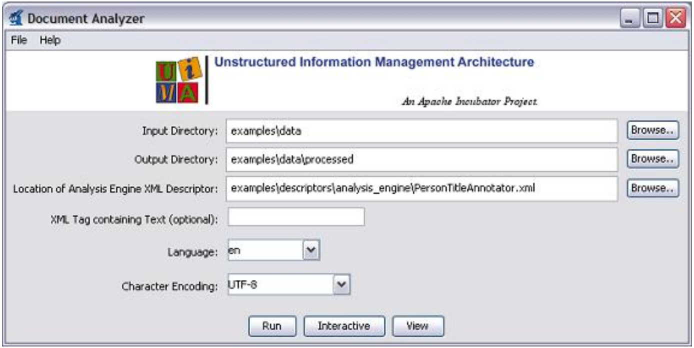
The AE code

```
package org.apache.uima.tutorial.ex1;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

import org.apache.uima.analysis_component.JCasAnnotator_ImplBase;
import org.apache.uima.jcas.JCas;
import org.apache.uima.tutorial.RoomNumber;

/**
 * Example annotator that detects room numbers using
 * Java 1.4 regular expressions.
 */
public class RoomNumberAnnotator extends JCasAnnotator_ImplBase {
  private Pattern mYorktownPattern =
        Pattern.compile("\\b[0-4]\\d-[0-2]\\d\\d\\b");

  private Pattern mHawthornePattern =
        Pattern.compile("\\b[G1-4][NS]-[A-Z]\\d\\d\\b");

  public void process(JCas aJCas) {
    // next slide
  }
}
```

The AE code (cont.)

```
public void process(JCas aJCas) {
  // get document text
  String docText = aJCas.getDocumentText();
  // search for Yorktown room numbers
  Matcher matcher = mYorktownPattern.matcher(docText);
  int pos = 0;
  while (matcher.find(pos)) {
    // found one – create annotation
    RoomNumber annotation = new RoomNumber(aJCas);
    annotation.setBegin(matcher.start());
    annotation.setEnd(matcher.end());
    annotation.setBuilding("Yorktown");
    annotation.addToIndexes();
    pos = matcher.end();
  }
  // search for Hawthorne room numbers
  // ..
}
```

UIMA Document Analyzer



Document Analyzer
File
Help
Unstructured Information Management Architecture
An Apache Incubator Project
Browse..
Browse..
Browse..
Run
Interactive
View

Input Directory: examples\data

Output Directory: examples\data\processed

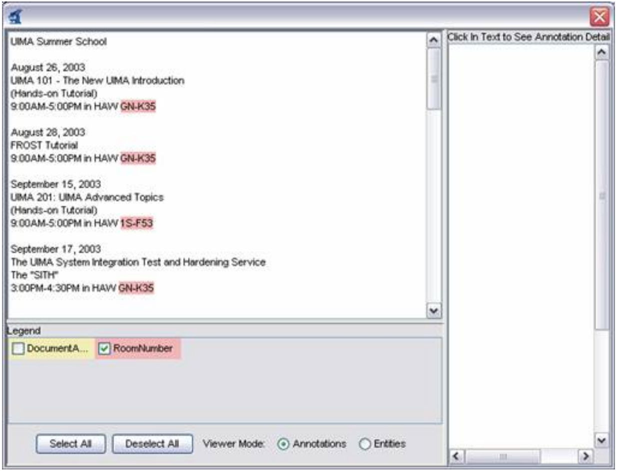
Location of Analysis Engine XML Descriptor: examples\descriptors\analysis_engine\PersonTitleAnnotator.xml

XML Tag containing Text (optional):

Language: en

Character Encoding: UTF-8

UIMA Document Analyzer (cont)



UIMA Summer School

August 26, 2003
UIMA 101 - The New UIMA Introduction
(Hands-on Tutorial)
9:00AM-5:00PM in HAW GN-K35

August 28, 2003
FROST Tutorial
9:00AM-5:00PM in HAW GN-K35

September 15, 2003
UIMA 201: UIMA Advanced Topics
(Hands-on Tutorial)
9:00AM-5:00PM in HAW 1S-F53

September 17, 2003
The UIMA System Integration Test and Hardening Service
The "SITH"
3:00PM-4:30PM in HAW GN-K35

Legend
☐ DocumentA... ☑ RoomNumber

Viewer Mode: ● Annotations ○ Entities

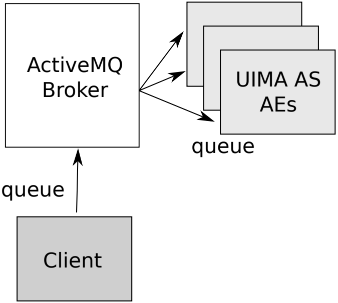
Custom Flow Controllers

- UIMA allows you to specify which AE will receive the CAS next, based on all the annotations on the CAS.
- `examples/descriptors/flow_controller/WhiteboardFlowController.xml`
  - FlowController implementing a simple version of the “whiteboard” flow model. Each time a CAS is received, it looks at the pool of available AEs that have not yet run on that CAS, and picks one whose input requirements are satisfied. Limitations: only looks at types, not features. Does not handle multiple Sofas or CasMultipliers.

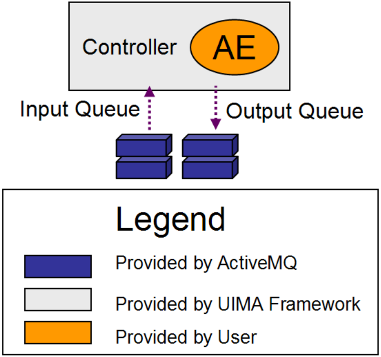
Outline

- Watson
  - Jeopardy!TM
  - Approach
- Apache Unstructured Information Management Architecture
  - Advantages
  - Mini-Tutorial
  - UIMA Asynchronous Scale-out (Low-latency)
- My Own Personal Contributions
  - To Watson
  - After Watson

UIMA AS: ActiveMQ



UIMA AS: Wrapping Primitive AEs



UIMA AS: Advantages

- Very flexible in terms of splitting the load between your nodes.
    - You have full control of how to split the queues into subqueues, etc.
- Very efficient in terms of network overhead.
    - A CAS to be split and processed multiple times (on different parts) is only sent once.
    - Only the **needed** annotations are sent and the **new** annotations are sent back.
        - Correct metadata (descriptor files) are key for that to work

UIMA AS: More information

- http://uima.apache.org/doc-uimaas-what.html
- http://svn.apache.org/viewvc/uima/uima-as/trunk/README?view=markup
- http://uima.apache.org/d/uima-as-2.4.2/uima_async_scaleout.html

Many frameworks

- Besides UIMA
  - `http://uima.apache.org`
- LingPipe
  - `http://alias-i.com/lingpipe/`
- Gate
  - `http://gate.ac.uk/`
- NLTK
  - `http://www.nltk.org/`

UIMA Advantages

- Apache Licensed
- Enterprise-ready code quality
- Demonstrated scalability
- Developed by experts in building frameworks
  - Not domain (e.g., NLP) experts
- Interoperable (C++, Java, others)

How Hard is to Learn UIMA?

- It is hard.
- Documentation is very good but very bulky.
  - Take the time to read it cover to cover, it goes down fast.
- Use the Eclipse tooling whenever possible.
- Learn uimaFIT first, then pure JCas, then CAS if needed.
- Focus on the “goodies”:
  - Apache UIMA Ruta – rule based annotators
  - OpenNLP – trained models for POS, NER, etc., easy to train your own
  - ClearTk – a wrapper around machine learning libraries

Outline

Watson

- Jeopardy!^{TM}
- Approach

Apache Unstructured Information Management Architecture

- Advantages
- Mini-Tutorial
- UIMA Asynchronous Scale-out (Low-latency)

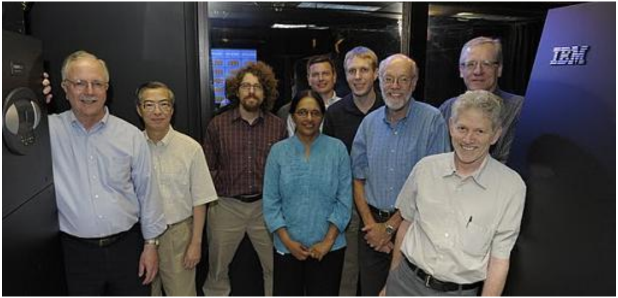
My Own Personal Contributions

- To Watson
- After Watson

My Contributions in the Watson System

- Sources Team
- Internal Tooling
- Machine learning in watson

Systems Team

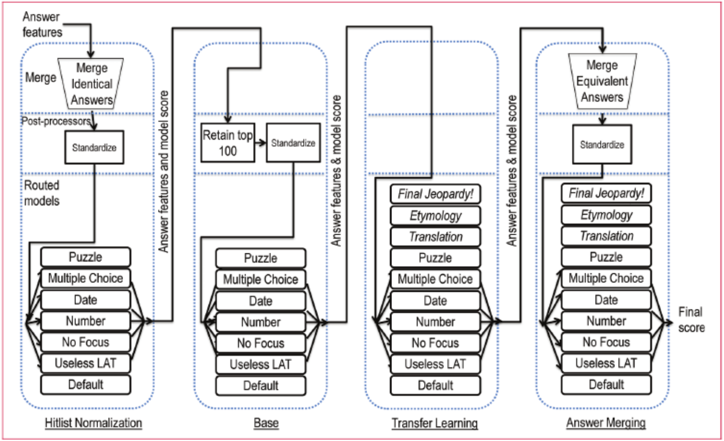


Systems Team, from `https://www.research.ibm.com/deepqa/`.

Machine Learning in Watson

- Multiple phases of Logistic Regression
- Feature Engineering
- DSL for Feature Engineering

First Four Phases of Merging and Ranking



from Gondek, Lally, Kalyanpur, Murdock, Duboue, Zhang, Pan, Qiu, Welty (2012)

Outline

- Watson
  - Jeopardy!$^{\text{TM}}$
  - Approach
- Apache Unstructured Information Management Architecture
  - Advantages
  - Mini-Tutorial
  - UIMA Asynchronous Scale-out (Low-latency)
- My Own Personal Contributions
  - To Watson
  - After Watson

After Watson

- Consulting
- Academic Work
  - Teaching
  - Hunter Gatherer
  - Thoughtland
- Free Software

Consulting

- MatchFWD: LinkedIn data
- UrbanOrca: Facebook data
- KeaText: legal data
- Radialpoint: tech support data
- Signed the open letter from Montreal tech community
  - `http://wearemtltech.ca`
- Contact me at `http://duboue.net`

Academic Work

- Taught a graduate-level course in NLG in Argentina.
- Published:
  - **Pablo Duboue**, Jing He and Jian-Yun Nie. *“Hunter Gatherer: UdeM at 1Click-2”*. NTCIR (2013).
  - Pablo Duboue. *“On the Feasibility of Automatically Describing n-dimensional Objects”*. EWNLG (2013).
  - Pablo Duboue. *Thoughtland: Natural Language Descriptions for Machine Learning n-dimensional Error Functions (demo)”*. Proc. of EWNLG (2013).
  - Jing He, **Pablo Duboue**, and Jian-Yun Nie. *“Bridging the Gap between Intrinsic and Perceived Relevance in Snippet Generation””*. COLING (2012).
  - Fabian Pacheco, **Pablo Duboue**, and Martin Dominguez. *“On The Feasibility of Open Domain Referring Expression Generation Using Large Scale Folksonomies (short paper)”*. NAACL (2012).
  - Pablo Duboue. *“Extractive email thread summarization: Can we do better than He Said She Said?”*. INLG (2012).
  - David Nicolas Racca, Luciana Benotti, and **Pablo Duboue**. *“The GIVE-2.5 C Generation System”* EWNLG (2011).

Hunter Gatherer

- What? 1-Click Search
  - Input: Query and 200 ranked Web pages
  - Output: a 1,000 characters summary
    - Summary should contain the information the pages relevant to the query.
- A research challenge part of NTICR
- Queries belong to 8 types (celebrities, how to, location, etc)
  - But the type is not explicit

Hunter Gatherer Approach

- Apply the DeepQA architecture to 1-Click task
  - Do not explicitly type the query
- Hunt nuggets, gather evidence
  1. Hunt text nuggets on relevant passages
  2. Gather evidence passages that contain nuggets and query terms
  3. Score nuggets based on evidence
  4. Final output are sentences containing highly scored nuggets

`https://github.com/DrDub/hunter-gatherer`

Thoughtland

- Generation of textual descriptions for *n*-dimensional data.
  - Early stage research
  - Focus on describing the error surface for Machine Learning models
- Presented at the European Workshop in Natural Language Generation in Sofia, Bulgaria (2013)
- Written in Scala, using Mahout on top of Hadoop for clustering and Weka for machine learning.
- Demo: `http://thoughtland.duboue.net`
- Code: `https://github.com/DrDub/Thoughtland`

Thoughtland: Input

- A small data set from the UCI ML repo, the Auto-Mpg Data:

`http://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/`

```
@relation auto_mpg
@attribute mpg numeric
@attribute cylinders numeric
@attribute displacement numeric
@attribute horsepower numeric
@attribute weight numeric
@attribute acceleration numeric
@attribute modelyear numeric
@attribute origin numeric

@data
18.0,8,307.0,130.0,3504.,12.0,70,1
14.0,8,455.0,225.0,3086.,10.0,70,1
24.0,4,113.0,95.00,2372.,15.0,70,3
22.0,6,198.0,95.00,2833.,15.5,70,1
27.0,4,97.00,88.00,2130.,14.5,70,3
26.0,4,97.00,46.00,1835.,20.5,70,2
```

... +400 more rows

Thoughtland: Output

- MLP, 2 hidden layers (3, 2 units), acc. 65%, Thoughtland generates:

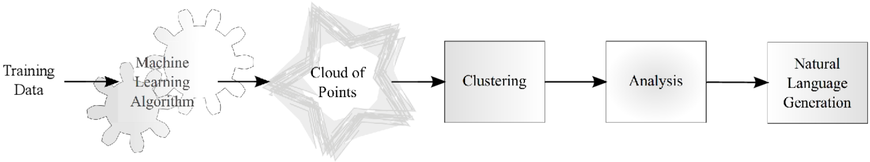
  > *There are four components and eight dimensions. Components One, Two and Three are small. Components One, Two and Three are very dense. **Components Four, Three and One are all far from each other.** The rest are all at a good distance from each other.*

- MLP, 1 hidden layer (8 units), acc. 65.7%, Thoughtland generates:

  > *There are four components and eight dimensions. Components One, Two and Three are small. Components One, Two and Three are very dense. **Components Four and Three are far from each other.** The rest are all at a good distance from each other.*

(difference is ***highlighted***)

Thoughtland: Architecture



Summary

- UIMA is a production ready framework for unstructured information processing.
    - It enables both batch processing and low latency processing.
- UIMA is a framework and it contains little or no annotators itself.
    - But new annotators are becoming available through OpenNLP and ClearTk.
- It is an efficient framework that requires commitment on behalf of its practitioners.
    - UIMA learning curve is fairly steep.