Extracting relational data from wikipedia

Chris Perivolaropoulos

Sunday 21 February 2016

Contents

1 Introduction

1.1 START

The START Natural Language System is a software system designed to answer questions that are posed to it in natural language. START parses incoming questions, matches the queries created from the parse trees against its knowledge base and presents the appropriate information segments to the user. In this way, START provides untrained users with speedy access to knowledge that in many cases would take an expert some time to find.

START (SynTactic Analysis using Reversible Transformations) was developed by Boris Katz at MIT's Artificial Intelligence Laboratory. Currently, the system is undergoing further development by the InfoLab Group, led by Boris Katz. START was first connected to the World Wide Web in December, 1993, and in its several forms has to date answered millions of questions from users around the world.

A key technique called "natural language annotation" helps START connect information seekers to information sources. This technique employs natural language sentences and phrases annotations as descriptions of content that are associated with information segments at various granularities. An information segment is retrieved when its annotation matches an input question. This method allows START to handle all variety of media, including text, diagrams, images, video and audio clips, data sets, Web pages, and others.

The natural language processing component of START consists of two modules that share the same grammar. The understanding module analyzes English text and produces a knowledge base that encodes information found in the text. Given an appropriate segment of the knowledge base, the generating module produces English sentences. Used in conjunction with the technique of natural language annotation, these modules put the power of sentence-level natural language processing to use in the service of multimedia information access.

2 Wikipediabase

2.1 Introduction

WikipediaBase base is a backend to START responsible for providing access to wikipedia related information. The WikipediaBase we refer to is a python rewrite of the now deprecated Ruby WikipediaBase.

2.2 People

The python implementation was initially written by Chris Perivolaropoulos and was eventually handed over to

- Alvaro Morales
- Michael Silver

2.3 Functionality

The WikipediaBase server provides operations to:

- find all classes that a Wikipedia term (object) belongs to, and
- given a specific wikipedia term and a class under which it functions, provide information

1. Queries
2. Ontology
   (a) Terms
   (b) Classes
3. Infoboxes

   Infoboxes are tables displaying information about a wikipedia page in a semi structured way. At the moment most of the data retrieved by WikipadiaBase comes from infoboxes

   {{TODO: example image}}

2.4 Getting started

1. Language

   The WikipediaBase implementation that we refer to is written in python. Previous implementations were written in Java and Ruby but the language of choice for the rewrite was python for multiple reasons:

   - Python is in the pre-graduate curriculum of MIT computer science department. This will ease the learning curve of new members of Infolab.
   - Python is a easy to learn and mature language with a rich and evolving ecosystem. This fact eases the introduction of new maintainers even further.

2. Getting the code

   The entire WikipediaBase resides in a git repository in infolab's github orginization page

   ```
   git clone git@github.com:infolab-csail/WikipediaBase
   ```

3. Virtualenv and dependencies

   WikipediaBase depends on multiple other python packages. Fortunately, python is shipped not only with a great package manager, but also with a mechanism called virtualenv that isolates installations of a project's dependencies from the rest of the system, thus avoiding problems like version or namespace collisions. The way this effectively works is that the global python installation is half copied half symlinked to a local directory and the dependencies are installed only in the local sandbox. To create and activate a python virtualenv:

   ```
   $ virtualenv --no-site-packages py
   $ . py/bin/activate
   $ which python
   /the/local/directory/py/bin/python
   ```

   Now that we can safely install anything we want without breaking any global installation

   ```
   pip install -r requirements.txt
   ```

We will need some extra stuff for WikipediaBase to work:

- Postresql
- Redis

The installation of these packages varies across platforms. Both these packages are databases. Their purpose is for caching and for storing ahead-of-time compitations like infobox markup name to rendered name matching.

4. Backend databases or live data

2.5 Architecture

1. Pipeline

   When resolving a query WikipediaBase employs a pipeline of modules to figure out what the best way to respond would be.

   (a) Frontend
   WikipediaBase can be used as a library but it's primary function is as a backend to START. The communication between START and WikipediaBase is carried out over a plaintext telnet connection on port {port} using EDN-like sexpressions. The frontend handles the network connection with START, translates the received queries into calls to knowledgebase and then translate the knowledgebase response into properly formulated sexpressions that it sends back over the telnet connection.

      i. Protocol

   (b) Knowledgebase
   The knowledgebase is the entry point to the rest of wikipediabase. It uses the Provider/Acquirer pattern to transaprently provide the frontend with arbitrary methods. Those methods are responsible for chosing whether we are to resort to classifiers or resolvers (or any other mechanism) for answering the query. Available classifiers and resolvers become accessible to the knowledgebase automatically using their base class.

   (c) Classifiers
   Each classifier is a singleton that implements a heuristic for assigning a class of an object. Thereare a couple classifiers available at the moment.

(d) Resolvers

Resolvers are also singletons but their purpose is to find the value of the requested property.

(e) Lisp types

Lisp type instances are wrappers for python objects or values that are presentable in s-expression form that START can understand. They are created either from the raw received query and unwrapped to be useful to the pipeline, or by the answer WikipediaBase comes up with and then encoded into a string sent over telnet to START.

2. Fetcher

   The fetcher is an abstraction over the communicatioln of WikipediaBase with the outside world. It is a singleton object that implements a specific interface.

3. Infobox

4. Article

2.6 Provider/Acquirer model

WikipediaBase attempts to be modular and extendible. To accomplish this it is often useful to have parts of the system that access resources (eg. heuristic methods) without knowledge of what module those came from. Additionally it is often the case that resources come from many different modules. To avoid ad-hoc code and hard dependencies the provider / acqirer model was created:

- Subclass provider/Acquirer classes
- The Provider uses the `@provide` decorator to provide resources.
- The acquirer has transparent access to the aggregate of provided values for a key.

2.7 Testing

1. Unit testing

   The good functioning of WikipediaBase is assured by a comprehensive test suite of unit tests, functional tests and regression tests.

(a) Unit tests

Unit tests test small blocks of functionality, that are composed to create the system at large. For unit testing we use python's default testing library. Each test is a class the subclasses

(b) Functional and regression tests

Functional tests are tests written before, during or shortly after the development of a system and they assert the correct overall functioning of the system. Regression tests are very akin to functional tests. They prove that a found bug was fixed and assert that it will not appear again later. Functional and regression tests currently reside in `tests/examples.py`

2. Examples

2.8 Synonyms

1. Good/Bad synonyms
2. Synonym generation

2.9 Backend databases

1. DBM
2. SQLite
3. Redis
4. Postgres

2.10 Data sources

2.11 Date parser

Dateparser resides in a separate package called overlay-parse

2.12 Parsing with overlays

The concept of an overlay was inspired by emacs overlays. They are objects that specify the behavior of a subset of a text, by assigning properties to it. An overlay over a text t in our context is tuple of the range within that text, a set of tags that define semantic sets that the said substring is a member

of, and arbitrary information (of type A) that the underlying text describes. More formally:

$$o_i \in TextRanget \times Set(Tag) \times A$$
$$Text \rightarrow \{o_1, o_2, ..., o_n\}$$

So for example out of the text

$$The weather today, \overbrace{Tuesday}^{o_1}\ \overbrace{21^{st}}^{o_2}\ of\ \overbrace{November}^{o_3}\ \overbrace{2016}^{o_4},\ was\ sunny.$$

We can extract overlays $\{o_1, ..., o_4\}$, so that

$$\begin{array}{lllll} o_1 = (& r(\text{"}Tuesday\text{"}), & \{\text{DayOfWeek, FullName}\}, & 2) \\ o_2 = (& r(\text{"}21^{st}\text{"}), & \{\text{DayOfMonth, Numeric}\}, & 21) \\ o_3 = (& r(\text{"}November\text{"}), & \{\text{Month, FullName}\}, & 11) \\ o_4 = (& r(\text{"2016"}), & \{\text{Year, 4digit}\}, & 2016) \end{array}$$

Notice how for all overlays of the example we have $A = \mathbb{N}$, as we encode day of the week, day of the month, month and year as natural numbers. We encode more precise type information (ie that a day is inherently different than a month) in the tag set.

Once we have a set of overlays we can define overlay sequences as overlays whose ranges are consecutive, that is their and their tag sets match particular patterns. For example we can search for sequences of overlays that match the pattern

$$p = \text{DayOfMonth}, \text{Separator}(/), (\text{Month} \wedge \text{Number}), \text{Separator}(/), \text{Year}$$

to match patterns like 22/07/1991, where $Separator(/)$ matches only the character "/"

2.13 The implementation

2.14 Optimization

1. Comparison

2.15 The dates example

2.16 Benchmarks

2.17 Future

1. Configuration
   (a) Persistence
   (b) Pass by reference
   (c) Lenses
   (d) Laziness
      i. Referential (Ref - Items)
      ii. Computational
2. START deployment
3. Test suites
4. Bugs
5. Answer hierarchy

3 WikipediaMirror

wikipedia-mirror is a tool for generating mirrors of wikipedia.org using the dumps provided by wikipedia.org.

3.1 Mediawiki stack

Wikipedia-mirror builds upon the mediawiki stack provided by bitnami. A service that builds the entire server within the confines of a directory. This is useful because we avoided the overhead of dealing with container or VM technologies and we had direct access to the filesystem of the stack while still having bitnami's build system do the tedious job of orchestrating the various components and separating our sever from the rest of the system.

The stack is comprised of

- An http server, in our case apache
- The web application runtime, in our case PHP
- A database, in our cas MySQL

- The web application itself, in our case mediawiki

All of the above are provided by the the bitnami mediawiki stack.

1. Extensions

   For mediawiki to act like wikipedia a number of extensions are required. The installation process of such extensions is not automated or streamline. To automatically manage this complexity a mechanism is provided for declaratively installing extensions. To add support for an extension to wikipediabase one needs to add the following code in `Makefile.mwextnesions` (modifying accordingly):

   ```
   MW_EXTENSIONS += newextension
   mw-newextension-url = url/to/new/extnesion/package.tar.gz
   mw-newextension-php = NewExtensionFile.php
   mw-newextension-config = '$$phpConfigVariable = "value";'
   ```

   And wikipedia-mirror will take care of checking if the extension is already installed and if not it will put the right files in the right place and edit the appropriate configuration files. The entry points for managing extensions are (provided that the name of the registered extension is newextension):

   ```
   make mw-print-registered-extensions # Output a list of the registed extensions
   make mw-newextension-enable         # Install and/or enable the extension
   make mw-newextension-reinstall      # Reinstall an extension
   make mw-newextension-disable        # Disable the extension
   make mw-newextension-clean          # Remove the extension
   ```

   All registered extensions will be installed and enabled when wikipedia-mirror is built.

3.2 Dumps

Wikipedia provides monthly dumps of all it's databases. The bulk of the dumps come in XML format and they need to be encoded into MySQL to be loaded into the wikipedia-mirror database. There are more than one ways to do that.

1. PHP script

   Mediawiki ships with a utility for importing the XML dumps. However it's use for importing a full blown wikipedia mirror is discouraged due to performance tradeoffs. Instead other tools like mwdumper are recommended that transform the XML dump into MySQL queries that populate the database.

2. mwdumper

   The recomended tool for translating the XML dumps into MySQL code is mwdumper. Mwdumper is written in java and is shipped separately from mediawiki.

   (a) Xml sanitizer

   (b) Article dropper

3.3 Automation

1. Makefiles / laziness
2. Shell scripts
3. Bitnami

3.4 Performance

1. Compile time

   Compile time includes the time it takes for:

   - Downloading all the components of a wikipedia server
     - The bitnami stack
     - mwdumper
     - mediawiki-extensions
   - Installing and building those components (~1 min)
   - Downloading the wikipedia dumps
   - Preprocessing the dumps (~10 mins)
   - Populating the mysql database (~10 days)

Builds were done on Infolab's Ashmore. The system's specs are quite high end but the bottleneck was the disk IO so less than 1% of the rest of the available resources were used during the MySQL database population.

(a) Attempts to optimizing MySQL

2. Runtime

   Runtime of wikipedia mirror turned out to be too slow to be useful and therefore the project was eventually abandoned. Namely for the full wikipedia dump of July 2014 the load time for the Barack Obama, not taking advantage of caching was at the order of ~30s.

4 Related CSAIL projects

5 This document

This document was written using emacs org-mode.