

M S RAMAIAH INSTITUE OF TECHNOLOGY

**The purpose for generating Question Papers applying data mining techniques in accordance with Bloom’s taxonomy.**

Author(s): Akshay C Shetty , Jagannathan G ,Rakshith S S ,Srivallabh B K

Guidance: Naresh E (Assistant Professor ,MSRIT)

Review of works: Naresh E(Assistant Professor ,MSRIT)

Published by : Department of Information Science & engineering, MSRIT

Copyrights:

Digital Security:

The purpose for generating Question Papers applying data mining techniques in accordance with Bloom’s taxonomy

1.1 Abstract

Automatic Generation of Question paper and statistical comparison on question papers created with and without the application Bloom’s taxonomy, considering the factors of Redundancy, Bloom’s protocols, magnitude of the keywords [1] present in the questions is the sole purpose of the Practical and Theoretical implementation of the project.

This tool is basically UI enabled question paper generator solution for Teachers, Tutors, Students and their Parents/Guardians, Schools and Coaching Institutes across classes and subjects. The software allows generating a question paper based on parameters like learning objectives, types of questions, competency level and difficulty level.

Emphasis has been given to tag each question with its learning objectives. The repository of questions allows teachers to select a variety of questions from the bank. The teachers have the flexibility to generate class tests, terminal tests, and final tests.

Setting a balanced question paper is a complex and demanding process. Generally most of the teachers set learning objectives while making lesson plans but they give less importance to transform these learning objectives into assessment objectives while setting question papers.

This peculiar feature of linking each question with its learning objective allows teachers to focus on the testing of desired outcomes of the learners. These products enable users to generate the question paper within a few minutes. It is therefore a very user friendly software solution.

According to the results of the experimental research it is possible to present the important conclusion: All Bloom’s cognitive goals of taxonomy domains are relevant and applicable enough.

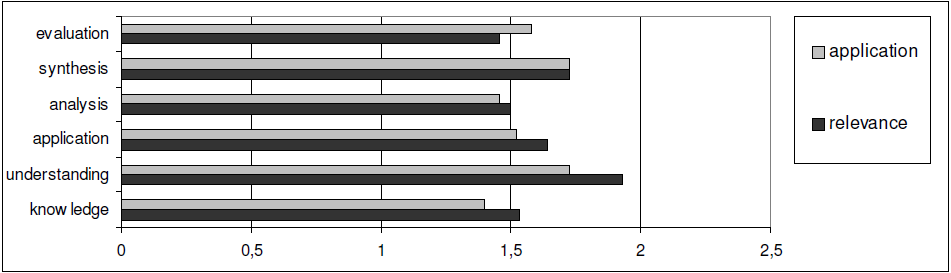


Figure 1.1

Fig1: With reference to the experimental evaluation analysis of data it is possible to propose that the main cognitive domains are evaluation, analysisand knowledge; the most applicable are knowledge, analysis and application. According to the experts, knowledge and analysis are relevant and applicable.

The standpoint of the respondents could be interpreted as follows: the cognitive domain “knowledge” is relevant and applicable, owning to the fact that without it the thinking it is impossible, i.e. without knowledge we are unable to educate the Capabilities of critical thinking.

They highlight the education of critical thinking because critical thinking cannot exist without evaluation. The least relevant criterion is “comprehension”. Unappreciated knowledge cannot exist, therefore, this experts’ opinion could imply their reference to reproductive pedagogy. Cognitive domains differ in their attitude of relevance and application; however, “synthesis” is equally relevant and applicable. The significance of relevance and applicability of other cognitive domains are out of synapse. Consequently, presumption can be made that the experts evaluate the relevance of the cognitive domains on Bloom’s Taxononmy.

1.2 Algorithm support and Technical Specification

**1.2.1 Natural Language Toolkit (Python)**

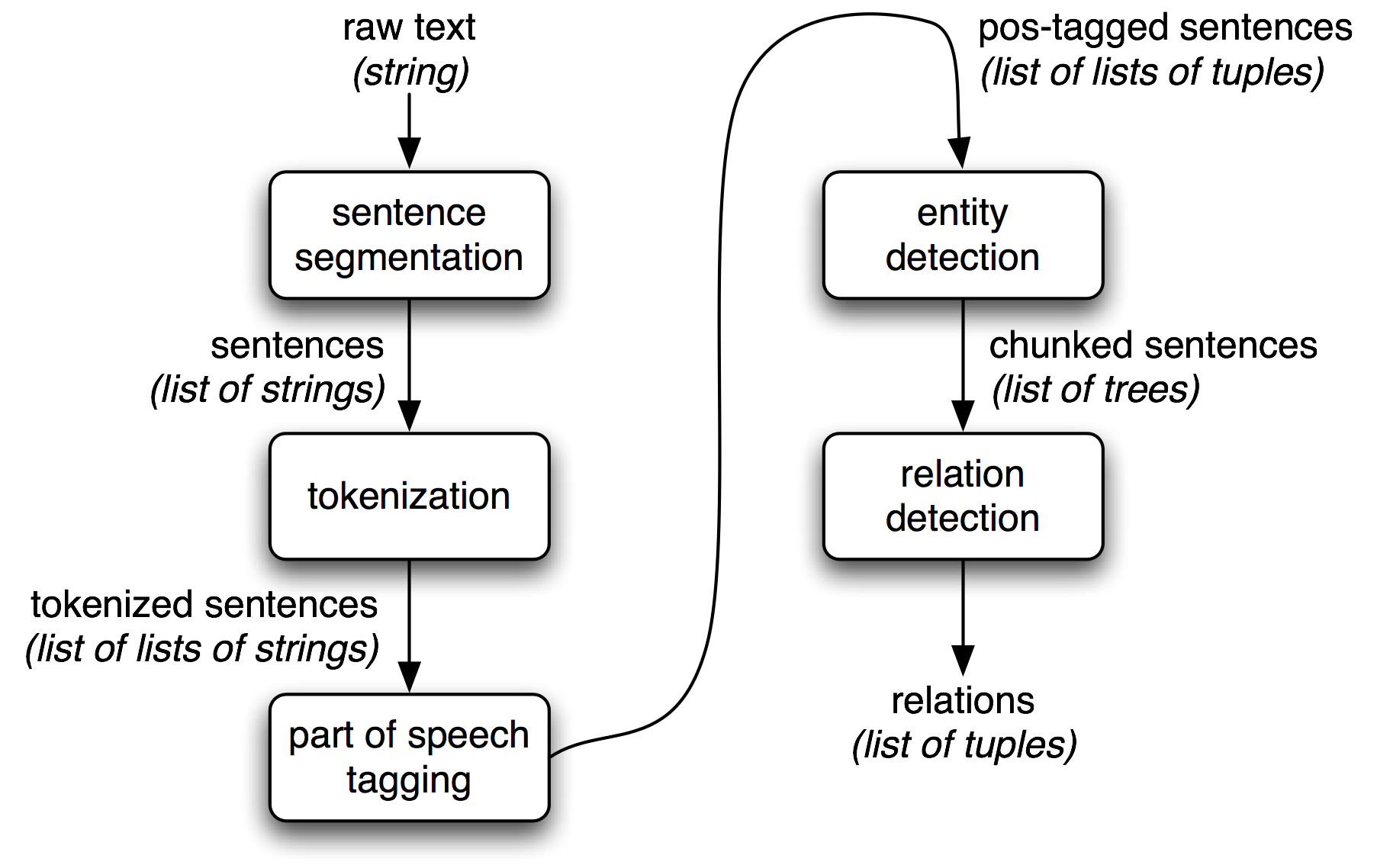
NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to [over 50 corpora and lexical resources](http://nltk.org/nltk_data/) such as WorldNet, along with a suite of text processing libraries for classification, tokenization, stemming, and tagging, parsing, and semantic reasoning.

[Natural Language Processing with Python](http://nltk.org/book) provides a practical introduction to programming for language processing. Written by the creators of NLTK, it guides the reader through the fundamentals of writing Python programs, working with corpora, categorizing text, analyzing linguistic structure, and more.

**1.2.2 Information Extraction from Text.**

For any given question, it's likely that someone has written the answer down somewhere. The amount of natural language text that is available in electronic form is truly staggering, and is increasing every day. However, the complexity of natural language can make it very difficult to access the information in that text. The state of the art in NLP is still a long way from being able to build general-purpose representations of meaning from unrestricted text. If we instead focus our efforts on a limited set of questions or "entity relations," such as "where different facilities located are," or "who is employed by what company," we can make significant progress. The goal of this chapter is to answer the following questions:

1. How can we build a system that extracts structured data, such as tables, from unstructured text?
2. What are some robust methods for identifying the entities and relationships described in a text?
3. Which corpora are appropriate for this work, and how do we use them for training and evaluating our models?

****

Information extraction architecture, Simple Pipeline Architecture for an Information Extraction System. This system takes the raw text of a document as its input, and generates a list of (entity, relation, entity) tuples as its output (Fig 1.2).

Fig 1.2 shows the architecture for a simple information extraction system. It begins by processing a document using several of the procedures. First, the raw text of the document is split into sentences using a sentence segmenter, and each sentence is further subdivided into words using a tokenizer.

Next, each sentence is tagged with part-of-speech tags, which will prove very helpful in the next step, named entity detection. In this step, we search for mentions of potentially interesting entities in each sentence. Finally, we use relationdetection to search for likely relations between different entities in the text.

# 1.2.3 Regular expression operations [re]

This module provides regular expression matching operations similar to those found in Perl. Regular expressions use the backslash character ('\') to indicate special forms or to allow special characters to be used without invoking their special meaning. This collides with Python’s usage of the same character for the same purpose in string literals which implicitly affects our purpose in this project; The solution is to use Python’s raw string notation for regular expression patterns; backslashes are not handled in any special way in a string literal prefixed with 'r'. So r"\n" is a two-character string containing '\' and 'n', while "\n" is a one-character string containing a newline. Usually patterns will be expressed in Python code using this raw string notation.

1.2.4 Tokenizing

**T**[**okenization**](http://en.wikipedia.org/wiki/Tokenization) is a way to split text into tokens. These tokens could be paragraphs, sentences, or individual words. NLTK provides a number of tokenizers in the [tokenize module](http://nltk.googlecode.com/svn/trunk/doc/api/toc-nltk.tokenize-module.html). Note that empty tokens are not returned when the delimiter appears at the start or end of the string. The material between the tokens is discarded.

1.2.5 Hash Map (Dictionary in Python)

A dictionary is mutable and is another container type that can store any number of Python objects, including other container types. Dictionaries consist of pairs (called items) of keys and their corresponding values.

Python dictionaries are also known as associative arrays or hash tables. The general syntax of a dictionary is as follows:

dict = {'Alice': '2341', 'Beth': '9102', 'Cecil': '3258'}

Each key is separated from its value by a colon (:), the items are separated by commas, and the whole thing is enclosed in curly braces. An empty dictionary without any items is written with just two curly braces, like this: {}.

Keys are unique within a dictionary while values may not be. The values of a dictionary can be of any type, but the keys must be of an immutable data type such as strings, numbers, or tuples.

2.1 Pyinstaller

PyInstaller is a program that converts (packages) Python programs into stand-alone executables, under Windows, Linux, Mac OS X, Solaris and AIX. Its main advantages over similar tools are that PyInstaller works with any version of Python since 2.4, it builds smaller executables thanks to transparent compression, it is fully multi-platform, and use the OS support to load the dynamic libraries, thus ensuring full compatibility. The main goal of PyInstaller is to be **compatible with 3rd-party packages out-of-the-box**. This means that, with PyInstaller, all the required tricks to make external packages work are already **integrated within PyInstaller itself** so that there is no user intervention required.

2.1.1 Features

* Packaging of Python programs into standard executables, that work on computers without Python installed.
* Multiversion: works under any version of Python from 2.4 up to 2.7.
* Explicit intelligent support for many 3rd-packages (for hidden imports, external data files, etc.), to make them work with PyInstaller out-of-the-box.
* Automatic support for binary libraries used through ctypes .
* Support for automatic binary packing through the well-known UPX compressor.
* Optional console mode (see standard output and standard error at runtime).
* Selectable executable icon.
* Fully configurable version resource section and manifests in executable.
* Support for building COM servers.

2.1.1 **Kivy**

* Kivy - Open source Python library for rapid development of applications  
  that make use of innovative user interfaces, such as multi-touch apps. Kivy runs on Linux, Windows, OS X, Android and iOS. You can run the same code on all supported platforms. It can use natively most inputs, protocols and devices including, Mac OS X Track pad and Magic Mouse, Mtdev, Linux Kernel HID, TUIO. A multi-touch mouse simulator is included.

Business Friendly - Kivy is 100% free to use, under an MIT license (starting from 1.7.2) and LGPL 3 for the previous versions. The toolkit is professionally developed, backed and used. You can use it in a commercial product.

The framework is stable and has a well documented API, plus a programming guide to help you get started.

Using Kivy on your computer, you can create apps that run on:

* Desktop computers: MacOSX, Linux, Windows.
* iOs Devices: iPad, iPhone.
* Android devices: Tablets, phones.
* Any other touch-enabled professional/homebrew devices supporting TUIO (Tangible User Interface Objects).