UAIC at GikiCLEF 2009

Adrian Iftene, Andrei-Cristian Prodan, Ion-Cătălin Condrea

UAIC: Faculty of Computer Science, "Alexandru Ioan Cuza" University, Romania
{adiftene, cristian.prodan, catalin.condrea}@info.uaic.ro

Abstract. This year marked UAIC[1]'s first participation at the GikiCLEF competition. For GikiCLEF 2009, systems needed to answer or address geographically challenging topics, on the Wikipedia collections, returning Wikipedia document titles as list of answers. The UAIC team's debut in this year competition has enriched us with the experience of developing the first system for the GikiCLEF task, at the same time setting the scene for next years participations. A brief description of our system is given in this paper.

1 Introduction

GikiCLEF[2] is an evaluation task under the scope of CLEF. Its aim is to evaluate systems which find Wikipedia entries/documents that answer a particular information need, which requires geographical reasoning of some sort. GikiCLEF is the successor of the GikiP[3] 2008 pilot task which ran in 2008 under GeoCLEF.

A system participating in GikiCLEF 2009 received a set of topics in all GikiCLEF languages (Bulgarian, Dutch, English, German, Italian, Norwegian, Portuguese, Romanian and Spanish) and it must produce a list of answers, in all languages it can find answers. The motivation for this kind of system behaviour is that in a real environment, a user prefers to read answers in his native language, but he is happy with answers (answers are titles of Wikipedia entries) in other languages he also knows or even just slightly understands.

GikiCLEF collections were represented by Wikipedia collections for all GikiCLEF languages and were available in three formats: HTML, SQL and XML. Participant systems used one of the versions of the collections and must offer answers to 50 topics prepared by organizers. In the end their answers have to point to valid HTML or XML files in the GikiCLEF collection.

The general system architecture is described in Section 2, while Section 3 is concerned with presentation of results. Last Section presents conclusions regarding our participation in GikiCLEF 2009.

[1] "Al. I. Cuza" University
[2] GikiCLEF: http://www.linguateca.pt/GikiCLEF/
[3] GikiP: http://www.linguateca.pt/GikiCLEF/index.php/Main_Page#GikiP_2008_pilot_task

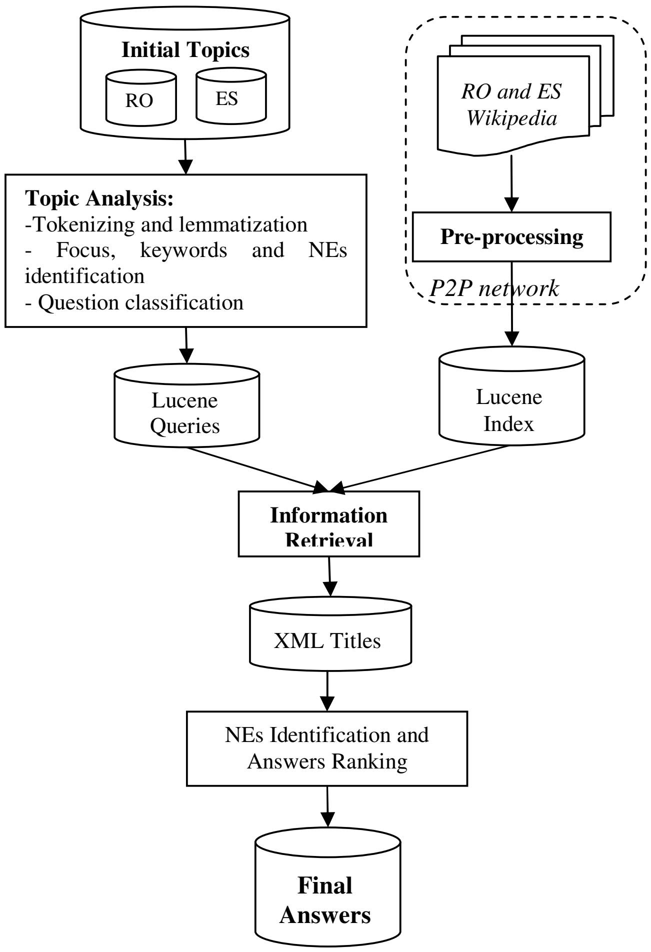

Figure 1: UAIC system used in GikiCLEF 2009

2 Architecture of the GikiCLEF System

The system contains four main modules that deal with corpora pre-processing, topic analysis, information retrieval and answers ranking (See Figure 1). For the pre-processing part we used a peer-to-peer network in which on separated computers we unzip initial XML files, pre-process them and after that we unify them in one common file. These files obtained on separated computers are afterwards sending to the indexing module. In what follow, we give few details and examples in order to understand better how our system works.

2.1 Corpus Pre-processing

From collections provided by organizers we used the XML version. Because, in the XML files a lot of tags were useless, we decided to eliminate these tags and to only keep relevant tags. This pre-processing part was done in two steps: in the first step we extract the relevant tags, and in the second step we eliminate from the content of tags identified at step 1, the formatting tags. The useful tags identified by us at step 1 were tags that contain paragraphs, titles, lines or columns from tables. At step 2 we eliminate tags for text formatting like bold, italic, underline, size, color, etc. and also the hyperlink tags.

For example, for file "*Active_Directory_3275.xml*" from English XML collection at first step one of the extracted paragraph tags was:

Table 1: Example of Paragraph Extracted after First Pre-Processing Step

```
<p id="wx8">
  <b id="wx9">Active Directory</b> (<b id="wx10">AD</b>) is an
implementation of <a
href="/wen/Lightweight_Directory_Access_Protocol"
title="Lightweight Directory Access Protocol"
wx:linktype="known"
wx:pagename="Lightweight_Directory_Access_Protocol"
wx:page_id="18508" id="wx11">LDAP</a> <a
href="/wen/Directory_service" title="Directory service"
wx:linktype="known" wx:pagename="Directory_service"
wx:page_id="334641" id="wx12">directory services</a> by <a
href="/wen/Microsoft" title="Microsoft" wx:linktype="known"
wx:pagename="Microsoft" wx:page_id="19001"
id="wx13">Microsoft</a> for use primarily in <a
href="/wen/Microsoft_Windows" title="Microsoft Windows"
wx:linktype="known" wx:pagename="Microsoft_Windows"
wx:page_id="18890" id="wx14">Windows</a> environments.
  Its main purpose is to provide central <a
href="/wen/Authentication#Computer_security"
title="Authentication" wx:linktype="known"
wx:pagename="Authentication" wx:page_id="47967"
wx:fragment="Computer_security" id="wx15">authentication</a> and
<a href="/wen/Authorization" title="Authorization"
wx:linktype="known" wx:pagename="Authorization"
wx:page_id="151617" id="wx16">authorization</a> services for
Windows-based computers.
  ...
</p>
```

And after the second step the same paragraph looks like in Table 2.

Table 2: Example of Paragraph Obtained after Second Pre-Processing Step

```
<p id="wx8">
  Active Directory (AD) is an implementation of LDAP directory
services by Microsoft for use primarily in Windows environments.
  Its main purpose is to provide central authentication and
authorization services for Windows-based computers.
  ...
  </p>
```

In this way we only keep the relevant text in new XML files.

2.2 Index Creation

The purpose of this module is to prepare the index necessary for retrieval of the relevant snippets of text for every topic. For this task we used the Lucene[4] indexing component. The index was created on the basis of the XML files obtained at the previous step. We have created one index at document level; in which, for fields, we insert the document title and all relevant text from a given XML.

2.3 Topic Analysis and Lucene Queries Creation

This step is mainly concerned with the building of Lucene query necessary in the retrieval part. Queries are created using the sequences of keywords, Lucene mandatory operator "+" and relevance operator "^" and "title" field. In this manner we obtain a regular expression for every question, which is then used in the search phase. In addition, it also provides the answer type, the question focus, and the question type. The topic analyzer performs the following steps (similar to [1]):

i) **NP-chunking** and **Named Entity extraction**,
ii) Identification of **question focus**,
iii) **The answer type** identification,
iv) **Inferring the question type**,
v) **Keyword generation**,
vi) Building of **Lucene query**.

For example for first topic "GC-2009-01" the output of this module is presented in Table 3. In tag <initial> is the initial form of the topic.

Table 3: Result Obtained after Topic "GC-2009-01" Analysis

```
<topic id="GC-2009-01">
  <initial>List the Italian places where Ernest Hemingway
visited during his life.</initial>
  <focus>places</focus>
  <verbs>list visited</verbs>
  <nouns>during life</nouns>
  <adjectives></adjectives>
```

[4] Lucene: http://lucene.apache.org/

```
  <nameEntities>Italian Ernest Hemingway</nameEntities>
  <lucene_query>(places^2 place) +Italian +Ernest +Hemingway
(visited^2 visit) during life (title:Italian title:Ernest
title:Hemingway  title:Italian Ernest Hemingway)</lucene_query>
  <answer_type>LOCATION</answer_type>
  <question_type>LIST</question_type>
</topic>
```

The meaning of the Lucene operators from Lucene query is the following:

- For first parenthesis "`(places^2 place)`", we search for word "*places*" or for word "*place*", but because "*places*" appears in the initial topic, this is more relevant (the boosting factor is 2) (by default, every word from Lucene queries has the value for boosting factor 1);
- "`+Italian`" means that is mandatory like text to contain this word;
- In "`title:Italian`" we specified that the search is done in the field "title";
- Between parentheses we have the default operator "or".

2.4 Answer Extraction

The purpose of this module is to retrieve from Lucene index created at 2.2 the relevant snippets of text for every topic, using Lucene query created at 2.3.

The building of list with final answers for every topic depends by Lucene score and by expected answer type. Thus, we calculate a new score based on score associated to every XML document retrieved by Lucene search engine and based on the correspondence between expected answer type and the type of named entities identified in the title of XML document. For example, if for a XML document we have difference between expected answer type and type of named entities identified in the title of this XML document, we insert a penalty in the new score.

For example, at topic "GC-2009-01", we identify the expected answer of type LOCATION (see Table 3 for details). In this case, we penalize the answers that don't contain a named entity of type LOCATION in the title and add additional points to the score of answers that contain one ore more named entities of type LOCATION.

3 Results

Our team submitted three runs. Details related to runs evaluation are presented in Tables 4 and 5. In Table 4 are presented the number of answers, number of correct answers and overall precision and score. In Table 5 are presented details related to Run 1, separated on each language. How we can see the best results were obtained on Spanish.

Table 4: UAIC Runs Details

	# answers	# corrects	Precision	Score
Run 1	6.420	8	0.0012	0.0156

	# answers	# corrects	Precision	Score
Run 2	1.133	2	0.0018	0.0062
Run 3	4.910	0	0.0000	0.0000

Table 5: Score per Language for Run 1

	BG	NL	EN	DE	IT	NO	NN	PT	RO	ES	Total
# answers	642	642	642	642	642	642	642	642	642	642	6.420
# correct answers	0	1	1	1	1	1	0	1	0	2	8
Precision	0	0.0016	0.0016	0.0016	0.0016	0.0016	0	0.0016	0	0.0031	0.0012
Score	0	0.0016	0.0016	0.0016	0.0016	0.0016	0	0.0016	0	0.0062	0.0156

4 Conclusions

This paper presents the UAIC system which took part in the GikiCLEF 2009 competition. The evaluation shows how the best score for our runs was 0.0156, and the best behavior was on Spanish language.

The system contains four components that deal with corpus pre-processing, index creation, topic analysis and answer extraction. In order to reduce the time necessary for pre-processing part we used a peer-to-peer network, in which this part was solved in a collaborative manner.

From our preliminary verifications we observe how the most errors were introduced by the answer extraction module, which was unable to extract correct answers. Another problem was encored during pre-processing part, when we observe how Romanian Wikipedia contains different encoding types for the same diacritics.

Acknowledgements

This paper presents a part of work of the Romanian team in the frame of the PNCDI II, SIR-RESDEC project number D1.1.0.0.0.7/18.09.2007. The authors would like to thank to the all students from group 3A, second year, for their help.

References

1. Iftene, A., Trandabăț, D., Pistol, I., Moruz, A., Balahur-Dobrescu, A., Cotelea, D., Dornescu, I., Drăghici, I., Cristea, D.: UAIC Romanian Question Answering system for QA@CLEF.In CLEF 2007. C. Peters et al. (Eds.), Lecture Notes in Computer Science, LNCS 5152, Pp. 336-343, Springer-Verlag Berlin Heidelberg, May (2008)