An Ontology-based Architecture for Natural Language Access to Relational Databases

Lawrence Muchemi[1], Fred Popowich[2]

[1]School of Computing & Informatics, University of Nairobi, Kenya
lmuchemi@uonbi.ac.ke
[2]Faculty of Applied Sciences , Simon Fraser University Burnaby, BC Canada
popowich@sfu.ca

Abstract. Natural language (NL) access to databases is a problem that has interested researchers for many years. We demonstrate that an ontology-based approach is technically feasible to handle some of the challenges facing NL query processing for database access. This paper presents the architecture, algorithms and results from the prototype thereof which indicate a domain and language independent architecture with high precision and recall rates. Studies are conducted for each of English and Swahili queries, both for same language and cross-lingual retrieval, from which we demonstrate promising precision and recall rates, language and domain independence, and that for language pairs it is sufficient to incorporate a machine translation system at the gazetteer level.

Keywords: Natural Language Interfaces, Databases, Ontologies

1 Introduction*

The problem of using natural language (NL) for database access is still a challenging problem. One specific challenge includes the lack of a suitable language- and domain-independent methodology for understanding unconstrained NL text in the context of a Swahili-English cross-lingual database. The cross-lingual aspect arises from the observation that most systems that support Swahili queries predominantly rely on concatenation of words or abbreviations in English as databases' metadata. Furthermore, the problem of parsing database schema into suitable ontology concepts that can readily map onto parsed NL text remains largely unstudied.

Attempts at machine learning methods have been reported in several reports including [8] and [19]. The main handicap has been the need for training data for each set of

* This Research was made possible by Foreign Affairs and International Trade Canada (DFAIT) funding through the Canadian Commonwealth Scholarship Program. It was also supported by a Discovery Grant from the Natural Sciences and Engineering Research Council of Canada.

database and NL queries and low performance in triple extraction. Other previous studies have tended to concentrate on logic based mapping of syntactically parsed NL to database contents but this has had the challenge of language dependence and brittleness due to reliance on rules and low database portability.

This paper presents a novel methodology that facilitates natural language understanding of user queries and that helps build structured queries that can be used to access highly structured information sources such as relational databases. We present a language and domain independent approach for understanding unconstrained NL text, show the design of our model and algorithms thereof that facilitate access of data from databases using Swahili and English as case-study languages. We also present results that were obtained from a prototype (developed as part of the wider project) upon which performance evaluations were done.

2.0 Related Work

Analysis of the related literature reveals the use of three general approaches: semantic parsing, logic mapping and ontology concept mapping.

In applications whose sources are characterized by low levels of structure such as web-based text sources the preferred approach is semantic parsing. Methods applied range from highly annotated machine learning techniques [11] to purely unsupervised techniques [6]. Though these efforts have not directly solved the access to relational database problem and only address question answering in text-based sources, some researchers notably Giordani and Moschitti [8] have taken this approach further by developing a parallel corpus of syntactic trees of NL queries and SQL queries for databases, specifically the Mooney GeoQuery set1. The main drawback to this approach is the need for training data for every set of database and NL queries pairs. This approach has therefore not been successful in solving database related problems.

NL access to relational databases has most successifully been tackled through logic mapping [16]. Logic mapping involves morphological and syntactic processing followed by mapping between the query syntactic forms and SQL forms. Usually semantic interpretation is achieved through semantic annotation of the identified phrases. Some successiful applications such as Tiscover [5] have been implemented through mapping of phrases. Tagging of phrases in a query with relevant predefined classes is necessary so that each phrase can be labeled and therefore semantically interpretated. Automatic tagging however inevitably introduces errors similar to those in semantic labeling such as poor classification while manual tagging is costly.

Due to these issues among others a different school of thought, encouraged by advances in the semantic web research, has emerged in search of improved models. A recent approach has been reported by the Nokia Research Centre Cambridge [15] where they solved the problem of accessing information stored in RDF repositories targeted to mobile phones users. These works along with other ground breaking works in the semantic web research such as AquaLog [18], Querix [7], NLP Reduce [9], QuestIO [17] and FREyA [4] though not directly solving the problem of relational

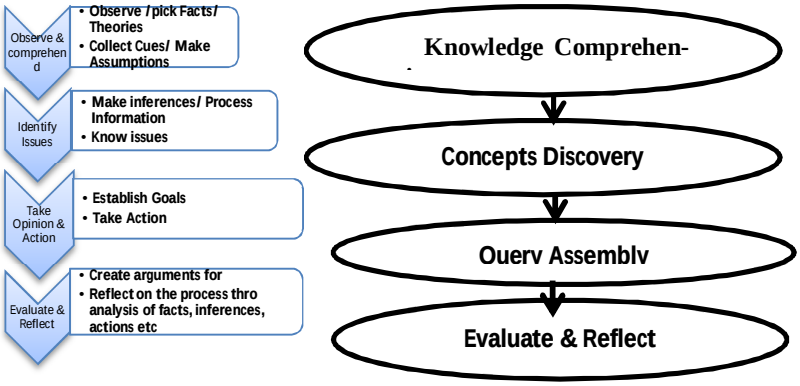
[1] *Geoquery* from University of Texas at Austin - AI Lab

databases have greatly inspired the approach in this work.

3.0 The Ontology-based Database Access Approach

Our approach, which we call Ontology Concept Mapping (OCM), is intuitively derived from the way human beings reason. Bond [2] has outlined a seven stage reasoning process. Levett-Jones [10] also outlines an eight step reasoning cycle resembling Bond's, except that he splits the observation phase into two. The two approaches are harmonized and shown in Fig. 1(a) below.

From this reasoning process one can reconstruct the human reasoning process specific to question answering by developing four layers of abstraction that map onto the seven steps in reasoning. Human beings first attempt to comprehend what the questioner is requesting. At times the information given is scanty or the questioner assumes we can derive more details from general knowledge. It is at this stage that humans attempt to derive reasonable additional knowledge in order to formulate the question fully and subsequently attempt to answer.



(a) (b)

Fig. 1. (a) Reasoning Process (Re-created from Bond (2011) and Levett-Jones (2009) and (b) Layers of Reasoning in OCM Approach

Subsequently we designed a four-layer abstraction model with layers for knowledge comprehension, knowledge discovery, query assembly, and evaluation and reflection as illustrated in the Fig. 1(b).

4.0 The Architecture

Through the use of feature space and gazetteer models, concepts from the NL query and database ontology are respectively represented and mapped via a matching function. A matching function is then applied to map equivalencies as illustrated in Fig. 2. A simple and effective method of knowledge comprehension and concepts

discovery is thus achieved. The triples are then pruned by eliminating triples not based on the composed semantic ontology of the database as these are less likely to yield results and ultimately assembled as SPARQL[2] queries that can be used to obtain information from Resource Description Framework (RDF) sources such as the semantic ontology for an RDBMS. The last step in Fig. 1(b) which is evaluating and reflecting on the answers provided is meant for self-assessment and automatic adjustment. This step was manually implemented as the algorithm's parameter adjustment and was not automated.

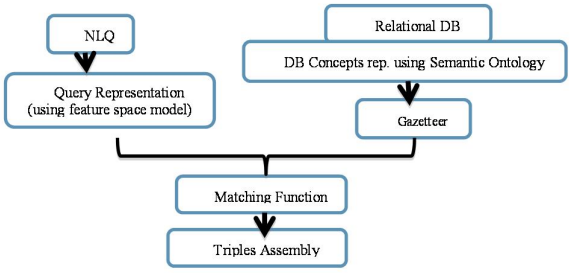


Fig. 2 Concepts Discovery

In the resulting architecture illustrated in Fig. 3 the system accepts user input in the form of full natural language sentences or key phrases or words. Raw text is subjected to linguistic processing that involves tokenizing, stemming, POS tagging and phrase formation based on Swahili collocation formation patterns established in [20]. On the other hand ontology elements relating to class and property names are normalized and stemmed. Instance names are stripped to the root as opposed to class and property which is stripped to the stem. The architecture works with the algorithm provided below.

The Algorithm

1. Assemble tokens list (words and phrases)

2. Comprehend ontology-strange terms that may be synonyms, hypernyms, hyponyms or even known jargon

3. Assemble List of Concepts (Tokens which match ontology elements)

- Explicit concepts

- Implicit concepts

- Concepts include matches to object and datatype properties, classes, instances, rdfs:labels, rdfs:comments and special categories (superlatives and enumeratives).

4. Assemble Triples by

- Determining Number of participating relations along with Primary and Foreign keys

-Identify User required properties and constraining instances and their related properties (Filters)

5. Assemble SPARQL Query

[2] SPARQL is a structured query language that can query RDF sources.

- Depending on discovered number of relations expand each query by heaping more and more triples until exhausted from user input.

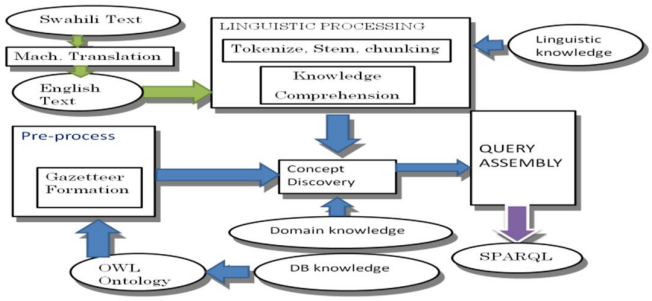


Fig. 3. Architecture for Ontology-based NL Access to DBs (ONLAD)

Cross-lingual Issues

One prime motivation behind this research was the quest for using Swahili to access data from databases. Swahili has over 150 million regular speakers in Eastern Africa. Unfortunately, nearly all databases are developed in a nation's official language which is predominantly English or French. This could be attributed to the fact that these official languages also double as the primary training languages and therefore database developers tend to favor their usage in database schema development. On the other hand, casual users tend to prefer using local languages or Swahili, the business languages among the communities. Thus a reasonable solution should have a cross-lingual nature to cater for these differing language usages.

We considered translation at two possible stages: at input (where the model subsequently uses English linguistic processing tools) and at the gazetteer level (where the model subsequently uses Swahili linguistic processing tools). It was noted that elements in a gazetteer are finite for a specific database and can be quickly translated accurately without too much overhead thus ensuring quality. So our system used translation at the gazetteer stage as opposed to NL input stage.

Parsing of DB Schema and NL Processing

In order to demonstrate how each of these layers work, let us consider the 'Northwind database'[3] which contains information about a company's inventory of its customers, suppliers and products sold. Furthermore, the customers can make orders which can be shipped by shippers whose information is also kept in the database. All information regarding employees is also maintained.

The section below shows how the knowledge comprehension, concepts discovery and query assembly processes occur. It has been shown that casual users prefer the use of short inputs. In some cases grammatical correctness of inputs is not a major priority [12]. Input to the model should therefore essentially be keywords but with capacity for accommodating users who prefer longer questions. At this stage of

[3] Microsoft. (2004). *Northwind Sample Databases for Microsoft SQL Server 2000*.

knowledge comprehension, the system can treat the user words as a bag-of-words and then naively match these words against database ontology similar to [9]. However on a closer look at how database authors represent relation-names and attribute names (equivalent to ontology class and property names respectively), we need an additional step to provide chunking on the input side and normalization on the ontology side. The preferred chunking is collocation.

Illustrative Examples

Given an input sentence in Swahili *Nipe majina, tarehe ya kuzaliwa na anwani za wafanyi kazi wote,* (which means '*Give me the names, dates of birth and addresses of all employees'*), we first translate the sentence using an external system (in our case the Swahili-English Google translator tool) to obtain an English equivalent.

A gazetteer, whose full architecture is described in [20] is constructed by identifying all legitimate class, property and instance names and arranges them as triples or class-property pairs. This can be done dynamically at run-time or in a preprocessing stage. Using preprocessing allows more effective translation at the gazetteer level. On the other hand NLQ is assembled in a feature space model whose architecture is described in [20]. The feature space model is a data structure which we implemented as an array and contains basic information of each word in a query such as the stem, hyponym, part of speech tag and other annotations. Conversion of relational databases into ontology was achieved through 'datamaster'[4].

The phrase *tarehe ya kuzaliwa* (translated to 'date of birth') mentioned above maps onto ontology element *{En: dateOfBirth}*. The normalized ontology element *dateOfBirth* yields three separate names; *date, of* and *birth.* It is therefore paramount that linguistic processing involves collocation chunking. These can now be matched against the user input. In order to enhance knowledge comprehension and thus usability capacity we used a stemmer. We used the Lancaster stemmer [13] as implemented in the Natural language toolkit [1] for English and a regular expression stemmer for Swahili. In the example above if the user used the word 'employed' instead of 'employees' which appears as an ontology concept, the system can intelligently conclude that we are referring to employees class — we have illustrated concept discovery. While this example involves the discovery of what we call "explicit" concepts in that they have direct mentions of properties or classes, in other scenarios there is no direct mention. Our approach provides for this implicit concept-discovery by performing simple inference.

Consider the question *Bidhaa gani ambazo huja kwa chupa (Which products come in bottles)?"* In this case the following processes occur according to our algorithm. *Bottles* is stripped to bottle which in turn maps to **instance** *bottled* which is found within the ontology as an instance. Since *Bottled* has been tagged in Gazetteer as an instance of *categories* class through datatype property *Description*, we discover two additional ontology concepts that is,

Categories **class** and

Description **property**

[4] *Plug-in for Importing Schemas and Data from Relational Databases into Protégé.*

and the following triple is created, where *FILTER* is necessary for instantiating a class's property value

?categories db: categories.description ?description
FILTER(?description = "bottled")

Further the query also has interrogative of type "which" that suggests an identification problem. By default we return *instances* of classes with properties related to identification of the class; that is *name, identification* or both if present in that particular class. We ultimately derive the following set of triples:

```
{ ?products db: products.ProductID ?ProductID.
   ?products db: products.ProductName?ProductName.
   ?products db: products.CategoryID ?CategoryID.
   ?categories db: categories.CategoryId ?CategoryID.
   ?categories db: categories.Description ?Description.
FILTER( ?Description = "bottled") }
```

Note that in database ontologies it is preferable to use both class and property names so as to minimize ambiguity in case multiple classes are using similar property names as in the example customer's phone and supplier's phone.

5.0 Experiments and Results

To evaluate a NL relational database access model we create a framework that takes into consideration precision, recall, support for multiple ontologies, portability and support for multiple languages and/or cross-lingual databases. In addition accuracy, F-score and 'effect of complexity of the question to precision and recall' were added to the evaluation criteria. The issue of complexity was handled in line with [5] and [17] where the complexity of a question is assumed to increase with the number of concepts present in a query.

Three sets of experiments were done. We applied the model across three databases namely farmers-db [12], Microsoft northwind-db and University of Nairobi students records database [20]. Each database was subjected to a set of randomly collected questions as described in each sources and the parameters as described above calculated. The farmers' database was queried with 625 randomly generated questions as described in [12]. For the Microsoft database we used a sample set of 120 questions, of which 100 were generated in [3] to evaluate the Elf system and 20 were generated in [12]. The third experiment for querying University of Nairobi students' records database had 310 unique questions generated in [20]. The databases which were originally authored in *MySQL* were converted into equivalent OWL ontologies through data-master plugin on protégé tool. The reasoner within the protégé tool was used to generate the answers from the ontologies. The generated answers were then manually evaluated to determine correctness and determine precision, recall and accuracy.

A full description of all results is beyond the scope of this paper. A summary is shown in Table 1 below. The results indicate a model whose average precision at a Levenshtein distance μ, of 1 (within the matching function) is 0.8393 and increases to

0.9480 on decrease of μ to 0. Precision therefore increases with a decrease of μ while recall decreases. Accuracy on the other hand increases slightly. A suitable parameter for gauging overall suitability would be the F-score, the harmonic mean of precision and recall, which increases from 0.7383 to 0.7525 on tightening μ. This means any NLQ system relying on string matching for extracting explicit and implicit concepts from a database should not permit too much laxity in the matching process

A high precision is important because it indicates the quality of the parsed queries while recall indicates the extent to which our model generates some SPaRQL queries whether right or wrong. [14] notes that all models decline to answer some questions hence the need for recall. Accuracy on the other hand indicates the extent to which a user expects the correct answer from a given set of questions.

Table 1. Summary of Results

	Levenshtein gap (match function)	**Exp. 1**	**Exp. 2**	**Exp. 3**	**Average**
Precision (%)	μ = 1	0.839	0.732	0.860	0.810
	μ = 0	0.948	0.936	0.945	0.943
Recall (%)	μ = 1	0.766	0.683	0.600	0.683
	μ = 0	0.646	0.650	0.584	0.627
Accuracy (%)	μ = 1	0.643	0.500	0.516	0.553
	μ = 0	0.613	0.608	0.552	0.591
F-Score	μ = 1	0.801	0.766	0.707	0.738
	μ = 0	0.769	0.707	0.722	0.753

Experimental results also indicate that translation is better done at the gazetteer level as opposed to the NLQ input level.

Minock *et al.* (2008) [21] has provided an elaborate review of performance of the most competitive models in logic-based mapping and semantic parsing approaches based on grammar mapping or statistical mapping, however a brief summary is provided in Table 2 below.

Table 2. Comparison of Results

Model	**Precision (%)**	**Recall (%)**	**Accuracy (%)**	**F-score**	**Main Principle**
PRECISE	0.80-0.100	0.550-.775	0.45-0. 775	0.65 - 0.87	Graph Matching [14]
WASP	0.800-0.915	0.600-0.940	0.500-0.866	0.690-0.930	Semantic parsing using Statistical Machine Translation (SCFG-based) [22]

Minock *et al*. Model	0.600-0.850	0.500-0.800	App. 0. 800	0.550-0.820	Synchronous Context-free with lambda calculus expressions (λ-SCFG) [21]
ONLAD (μ = 1- 0)	0.810-0.943	0.683-0.627	0.553-0.591	0.738-0.753	OCM Approach [this paper]

The model developed here can support cross-lingual databases. It has high portability and is not affected by distribution drift' because it does not require prior training unlike semantic parsing systems which rely on machine learning. The model is not affected by long ungrammatical questions nor complex questions with multiple concepts because it rarely relies on syntactic parsing information except for collocation formation. The model however suffers from lower maximum recall and accuracy levels as evident from results because it requires someone to enter information that sometimes is regarded as obvious or superfluous. For example the query 'give me customers who come from Nairobi' might require you to add the word 'name' within the query so that the system realizes we require 'customers' names'. The model can also be easily interchangeable between languages because only the collocation rules need to be imported and the appropriate gazetteer translator installed.

Conclusion

This paper has presented a language and domain-independent 'ontology concepts mapping' model that converts NLQ into structured queries that can be used to access relational databases. The model is especially suitable for resource-scarce languages with minimal linguistic processing activities. The model capitalizes on the easy conversion of NLQ to concepts which are processed as collocation chunks, and conversion of database schema and data as ontology concepts which are then mapped and assembled as SPaRQL queries. We have demonstrated how to overcome cross-lingual issues in database querying. We have successfully applied the model across three databases namely farmers [12], Microsoft northwind and UoN MSc Coordinator [20] where the average precision is 0.87.

References

1. Bird C., Loper. E., & Klein, E. (2009). *NLP with Python-Analyzing Text with the Natural Language Toolkit.* O'Reilly Media Inc.
2. Bond, T. (2011). *The Reasoning Process*. Retrieved 2011, from ICT New Zealand: http://ictnz.com/Thinking Pages/reasoning.htm
3. Bootra, R. (2004). *NLI: Comparing English Language Front End and English Query*. Master's Thesis, Virginia Common Wealth University,USA.
4. Damljanovic, D., Agatonovic, M., & Cunningham, H. (2010). *NLI to Ontologies: Combining Syntactic Analysis and Ontology-based Lookup through the User Interaction.The Semantic Web: Research and Applications. 106-120*
5. Dittenbach, M., Dieter, M., & Helmut, B. (2003). A NL Query Interface for Tourism Information.*ENTER*,10th Int. Conference on IT in Tourism 152-162.

6. Poon, H., &. Domingos, P. (2009). Unsupervised Semantic Parsing. *EMNLP*, (pp. 1-10). Singapore.
7. Kaufmann, E., Bernstein, A., & Zumstein, R. (2006). Querix: A NLInterface to Query Ontologies based on Clarification Dialogues. *International Semantic Web Conference (ISW 2006).* Athens, Georgia-USA: ISW2006.
8. Giordani, A., & Moschitti, A. (2010). Corpora for Automatically Learning to Map Natural Language Questions into SQL Queries. *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC'10)* (pp. 2336-2339). Valletta, Malta.
9. Kaufmann, E., Berstein, A., & Fischer, L. (2007). NLP-Reduce: A "naive" but Domain Independent NL Interface for Querying Ontologies. *4th European Semantic Web Conference (ESW2007).* Innsbruck-Austria.
10. Levett-Jones and Project Team. (2009). *Clinical Reasoning.* Retrieved 2011, from Instructor Resources.
11. Mooney, R. (2007). Learning for Semantic Parsing. *Eighth International Conference on Computational Linguistics and Intelligent Text Processing* (pp. 311-324). Mexico City, Mexico: Springer.
12. Muchemi, L. (2008). Towards Full Comprehension of Swahili NL for Database Querying. *Strengthening the Role of ICT in Development* (pp. 50-58). Kampala-Uganda: Fountain Publishers.
13. Paice, C. (1990). Another Stemmer. *ACM SIGIR*, 56-61.
14. Popescu, A., Etzioni, O., & Kautz, H. (2003). Towards a Theory of Natural Language Interfaces to Databases. *2003 International Conference on Intelligent User Interfaces.*, (pp. 149-157).
15. Ran, A., & Lencevicius, R. (2012). *Natural Language Query System for RDF Repositories.* Retrieved March 12, 2012, from http://alumni.cs.ucsb.edu/~raimisl/SNLP.camera.pdf
16. Shin, D.-G., & Chu, L.-Y. (1998). Establishing Logical Connectivity between Query Key Words and Database Contents. *12th Biennial Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence.* UK: Springer-Verlag London.
17. Tablan, V., Damljanovic, D., & Bontchev, K. (2008). A Natural Language Query Interface Structured Information. *5th Annual European Semantic Web Conference (ESWC).* Ternerife-Spain.
18. Lopez, V., Pasin, M., & Motta, E. (2004). AquaLog: An Ontology-Portable Question Answering System for the Semantic Web.
19. Zettlemoyer, L., & Collins, M. (2005). Learning to Map Sentencies to Logical Form. *Twenty First Conference on Uncertainty in Artificial Intelligence* (pp. 658-666). Edinburgh, Scotland: AUAI Press.
20. Muchemi, L. (2012). *NL Access to Relational Databases: An Ontology Concept Mapping (OCM) Approach.Nairobi, Kenya*: Unpublished PhD Doctoral Thesis, University of Nairobi.
21. Minock, M., Olofsson, P., & Naslund. (2008). Towards Building Robust NL Interfaces to Databases. *13th Int. Conference on Applications of NL to Information Systems, London, UK. 5039*, pp. 187–198. Hiedelberg: Springer.
22. Ge, R., & Mooney, R. (2005). A statistical Semantic Parser that Integrates Syntax and Semantics. *CoNLL0-5*, (pp. 9-16). Ann Arbor - Miami.