QolA: Fostering Collaboration Within QA

Diana Santos and Luís Costa

Linguateca, Oslo node and SINTEF ICT, Norway
Diana.Santos@sintef.no, Luis.Costa@sintef.no

Abstract. In this paper we suggest a QA pilot task, dubbed QolA, whose joint rationale is allow for collaboration among systems, increase multilinguality and multicollection use, and investigate ways of dealing with different strengths and weaknesses of a population of QA systems. We claim that merging answers, weighting answers, choosing among contradictory answers or generating composite answers, and verifying and validating information, by posing related questions, should be part and parcel of the question answering process. The paper motivates these ideas and suggests a way to foster research in these areas by deploying QA systems as Web services.

1 Motivation

There were many reasons that led us to propose QolA in the CLEF 2006 workhop in Alicante [18], which we would like to expand and further motivate here. Some are related to CLEF's ultimate aims as we understand them: namely advance the field of crosslingual IR systems and provide a forum for researchers and developers to cross the cultural barrier of dealing only with their own language, especially in a European context. Even though we believe that it is rewarding in itself to be together with QA researchers from some 40 groups who together process 8 languages, we are convinced that a lot can still be done to improve cross-fertilization of the different approaches and expertise that has come together (for example, hardly any paper in the QA section mentions other QA papers in CLEF – especially not those dealing with other languages).

Other goals are related to the QA field in itself and our attempt to advance it, inspired by the suggestions in Burger et al.'s roadmap [5], and in Maybury [15] and Strzalkowski and Harabagiu's [20] recent books (both of them, however, devote very little attention to crosslingual or multilingual issues). CLEF was the best place to do this because of the many collections in different languages it provides, and because it includes Portuguese. (Linguateca's aim is to significantly improve the state of the art of Portuguese processing).

1.1 Going Beyond the "1-1 Model"

Question answering can be seen from two complementary perspectives, to which we refer to as the information retrieval (IR) and the artificial intelligence (AI) ones, inspired by Wilks's analysis [24]:

C. Peters et al. (Eds.): CLEF 2006, LNCS 4730, pp. 569–578, 2007.

– QA is more user-friendly information retrieval, for less computer-literate users;
– QA is (one of) the best application(s) to test a system's understanding.

Not surprisingly, depending on one's view on the ultimate meaning of QA (what is QA really about), efforts by QA developers and researchers are devoted to different endeavours:

On the one hand, the IR tradition takes if for granted that QA is about massaging the documents that are returned by IR, so that QA can be approximated into a translation of questions into query terms, with subsequent choice of the appropriate passage (or part of it) for the answer. The TREC setup [21] and Brill et al. [4] are good illustrations of such an approach; the lexical chasm model of Berger et al. [3] is probably the cleverest way of solving QA as IR.

On the other hand, the AI view of QA as testing the understanding of a system leads researchers to ellaborate strategies of encoding knowledge and reasoning about it. Research under this paradigm is invariably concerned with understanding why the question was posed and what is expected of the answer (with the standard reference dating back to Lehnert [12]).[1]

One can make here the analogy with, in the fields of summarization and speech synthesis, extractive or concatenative paradigms as opposed to generation or synthetized ones: the first approach seeks primarily to *find* something that is already there, while the second *creates* something from the raw material.

For both approaches, however, there is one common difficulty: a question-answer pair is basically describable as a M,N-O,P relationship between queries and answers: There are many ways (M) to pose the same question, and there are many questions (N) that get the same answer. Plus: there are many correct ways to give the same answer (O), and many correct answers that can be given to the same question (P). So, one essential problem in open-domain QA is, given a particular question, to provide or choose one particular answer (or an acceptable combination of several ones). Taking one step further, [11] even argue for an approach of considering different candidate answers as potential allies, instead of competitors.

Technically, this means that, no matter their specific perspective for the solution, (a) dealing with alternative answers, (b) dealing with alternative ways of posing questions, (c) ranking different answers and (d) combining them, are tasks relevant to every QA system.

Now, one way to handle a multitude of related questions and answers is invoking a population of automatic question answerers.[2] This way, one can not only gather several different (or similar) answers to a same question, but also pose similar or related questions, as suggested e.g Wu and Strzalkowski [25]. This is

[1] Although this is not necessarily the only AI/NLP origin of QA: Dalmas and Webber [11], for instance, refer both to the natural interaction with databases and to the reading comprehension task as precursors of present-day "QA over unstructured data".

[2] A similar approach, invoking software agents, was first implemented and proposed by Chu-Carroll et al. [8], who report very positive results.

the main rationale for the QolA setup, which, as a by-product, also aims to stimulate the collaboration among different approaches that use different resources in different languages and search in different collections.

1.2 Real Answers Are Usually Not Found in One Place Only

In addition to the general motivation above (the M,N-O,P model), most answers to a real question are often crucially dependent on the user context and intentions. So, we believe that emphasis should be put to harvesting sets of possible answers and defining methods to provide useful and informative answers instead of betting on the one-answer only model.[3]

In fact, as early as TREC 99 [21], with the apparently unequivocal question *Where is the Taj Mahal?*, it became clear that correctness of an answer could not be decided beforehand, and that much more useful than an answer like *Agra* would either be some tourist information on how to get there (e.g., *half an hour by train from New Delhi*) or the address of the Taj Mahal casino in New Jersey. Not only proper names can have many referents, though: even question words such as *where* in *where was X stabbed?* can be interpreted in radically different ways – and thus the equally relevant answers *in the neck* and *in Cairo* – as well as common concepts such as *football*: which kind? [17]. Also *where*-questions depend crucially on the context, so that they can even sidetrack a general semantic type-checking approach, as Schlobach et al.'s [19] discussion of *where is hepatitis located?* shows.

Furthermore, apparently neutral questions depend on both the answerer and questioner's standpoints, and it is arguably better to provide more than one standpoint and leave the questioner to choose. For a "simple" question such as *Who was Freud?* we can get from the Web both *Freud was a terrible sadist who had the mood of a hysterical woman* and *Freud was a born leader with a deep knowledge of humanity and unerring judgment.* It is probably more interesting for most people interested in Freud to get a fair idea of the reactions to him than have an automatic system perform a choice for us.

In 2006 a substantial improvement was done to the QA@CLEF setup by accepting, not only a set of justification snippets per answer but a set of possible (ten) answers for each question as well. However, this new feature was apparently not used by most systems, which continued to send one answer only, probably because of the way they are conceived.[4] QolA would provide systems with the possibility to experiment with any number of answers, as well as with strategies to concatenate and/or decide on the final set. Instead of the quest for the only unique answer, QolA was therefore thought to help experiment with merging answers, weighting answers, choosing among contradictory answers or generating

[3] That this is a hopelessly wrong conceptual model is clearly displayed in [11]'s quantitative estimate of the number of distinct correct answers for TREC questions, which were originally conceived under the question-with-one-answer-only model.

[4] To be fair, we should also mention that most questions provided by the QA@CLEF organizers were still of the one-correct-answer-only variety, so there was no real incentive at last year's CLEF to return multiple answers.

composite answers, and verifying and validating information, by posing related questions. All these may be parts of the answer.

1.3 Going Beyond "One System at a Time" and Closer to the User

In the QA@CLEF evaluation setup, each system mainly competes with the others. Even though the CLEF spirit is crosslingual, most work so far [14] has been done on monolingual QA (each language has its own collection(s)), and at most bilingual retrieval: the joint use of more than one language collection has never been tried.

One of the possible drawbacks of QolA is that, in a QolA run, no single system would get the honour or responsibility of the answer. While this may diminish the individual zest of developers to provide the best individual system, we believe that, in a realistic setup, a user is not ultimately interested in how the answer has been arrived to, which system got it right, and which collections have been searched, provided s/he gets the source information (the justifications) and can confirm or discard the answer.

Ever since FAQFinder [6], there are growing repositories of previous answers and systems that mine them, so it is not realistic that future systems will depend only on their own proprietary bases created from scratch. On the contrary, harnessing the common knowledge and making use of many sources of information is key to the Web 2.0 perspective [16].

2 How to Operationalize Collaboration

The first challenge in QolA was to provide a way to invoke the several participating QA systems, and to us the best solution would be that the participating QA systems were available as web services (WS). This was still not enough: the services had to be interoperable as well. If people defined their systems separately, they would most likely provide operations with different names, different parameters and parameter names, etc. Therefore, a task of the organization was to define a set of operations relevant for QolA, which all providers of QA systems for QolA had to obey.

These operations were specified using WSDL (web service description language), [23] which is a de facto standard for describing web services. A WSDL web service definition provides information about the names of operations and parameters, but not about the functionality of the service. There are several research initiatives which aim to define the semantics of web services (WSDL-S, OWL-S, WSMF, IRS) [2,7], but for this first edition of QolA we decided to keep it simple and only cater for syntactic interoperability.

2.1 Web Service Population

The participating QA systems in QolA should be available as web services according to the specification that we proposed (and which evolved considerably with the feedback from the potential participants).

At some point, Sven Hartrumpf from the University of Hagen suggested the inclusion of translation web services in addition to the QA web services. Even though we were expecting the participation of bilingual QA web services and had already stipulated that we would provide every question in all QolA languages, we quickly agreed that separate translation Web services would make a lot of sense in the context of QolA and CLEF in general.

It was therefore arranged that there would be two kinds of Web services in QolA: QA services and machine translation (MT) services, provided by (special) participants called **QolA providers**.

General participation in QolA would be granted to anyone interested in invoking these services, and this kind of participants were dubbed **QolA consumers**.

2.1.1 QA Web Service Definition

The QA web services should provide the following set of basic operations: *GetSupportedLanguagePairs*, *GetSupportedEncodings*, *GetQuotaInfo*, *GetResourcesUsed*, *GetModulesUsed*, *GetAccuracy* and *GetAnswer*:

GetSupportedLanguagePairs: This operation returns a list with the language pairs that the QA system can handle. For example, questions in Portuguese/answers in Spanish, questions in English/answers in Italian, etc.

GetSupportedEncodings: This operation returns a list with the character encodings accepted by the QA Web service (`UTF-8`, `ISO-8859-1` or others).

GetQuotaInfo: It would be natural that QA web services had some mechanism to limit the number of requests per customer, not only for internal performance issues, but also to give a fair chance to all participants using the web service. GetQuotaInfo gives information about the available quota for a customer (how many questions can be asked to the QA web service for a certain period).

GetResourcesUsed: This operation returns a list of the resources employed by the QA system. This should include a list with names of ontologies, gazetteers, specific publicly available collections that are used by the system, such as `WordNet 6.1`, `CHAVEanot`, `DutchWikipedia`, `TGN`, etc. (These and other codes would have to be agreed for QolA.) This list is important for a consumer to decide whether a same answer returned by different systems came from the same source or from different ones.[5]

GetModulesUsed: This operation returns a list with the internal modules used by the QA system (`CoreferenceResolution`, `DeixisResolution`, `InferenceEngine`, `AnswerValidation`, `NaturalLanguageGeneration`, `NamedEntityRecognizer`, `Parser` and other codes agreed among the participants and organizers of QolA).

GetAccuracy: The idea behind this operation was to provide consumers with some information about the performance of a particular QA web service for

[5] However, [8] make the important point that, even when two answering agents – in our case, two different systems – consult the same sources, they may arrive at an answer with significantly different processing strategies, which is a (partial) motivation for the next operation, *GetModulesUsed*.

a certain language pair. For example, the system could return accuracy for the 2006 questions in general, or for each kind of question. No consensus as yet has been reached as to obligatoriness and format of this operation (note that systems could have improved dramatically since 2006, or not have participated in that event).

GetAnswer: This operation receives a question, the language and character encoding of the question, the language of the desired answer and a customer code (that could be used to limit the number of allowed questions per customer). The answer would consist in a string with the answer itself, a confidence score, the character encoding of the answer, the list of resources used, the list of modules used, and a list of justifications. Each of the justifications contains a document ID, a snippet of the document and a confidence score for the justification. Note that even if the system were not able to answer, a NIL answer, with a confidence score, should always be provided.

2.1.2 MT Web Service Definition

The MT web services should provide the following set of basic operations: *GetSupportedLanguagePairs*, *GetSupportedEncodings*, *TranslateQuestion*, *TranslateNamedEntity* and *TranslateSnippet*:

GetSupportedLanguagePairs, GetSupportedEncodings: These operations return lists with the language pairs and encodings that the MT system can handle.

TranslateQuestion, TranslateNamedEntity, TranslateSnippet: These three operations are provided because different techniques can be used to translate questions, named entities and text snippets. All of these operations receive the text to be translated, its language and character encoding and the target language for the translation. The result is the translation of the source text to the target language and the character encoding of the translation.

2.2 The Task

In order to make the least changes, we decided to use exactly the same evaluation setup that would be used (and therefore organized) for QA@CLEF 2007 main task. So the evaluation of the QolA results would be done the very same way the other sets of answers were dealt with by CLEF QA assessors. The way participants used the combination of the several WS available would make all the difference, we hoped.

In order to allow the future consumers to experiment with the full QolA concept, and considering that there might be a period where QolA providers had not yet made their systems available, Bernardo Magnini suggested the creation of a set of fake services which might only answer previous questions, but which could be made available by the organization, provided previous QA@CLEF participants allowed us to use their previous runs (and most at once did).

2.3 Baselines and Runs

To participate in QolA, a consumer participant would have to provide a logic, in the form of a program, for invoking the different services available.

In order to encourage participants to really try collaboration, the organization should provide a large variety of simple baselines, such as: find the quickest answer, random choice, majority choice, confidence score choice, intersection and union choices, etc., so that participants were "forced" to do something better.

Also, QolA's organization disallowed "selfish" runs, in the sense of runs centered in one's own system. This means that fixed schemas invoking any system by name were forbidden – only general runs were allowed, to prevent strategies like "let me first invoke my system, then the best of last year's CLEF...": No system should be given advantage that was not based on its own current behaviour.

3 Expected Results

(Consumer) participants were expected to experiment in QolA with at least the following issues:

- validation of previous answers in other collections (possibly with yes/no questions), or with different but synonymous formulations (see Clarke et al. [9] and Magnini et al. [13] on validation approaches and exploiting redundancy for this purpose);
- mining related questions and related information on the same subjects (as advogated in [11]);
- integrate in different ways related information (including both Chu-Carroll et al's [8] "answer resolution" and the "answer tiling" of Ahn et al. [1]);
- use redundancy in collections or across collections to rank answers (see i.a. Clifton et al. [10]);
- devise strategies to separate same answers from really different answers;
- try to get some grasp of getting different answers in different contexts;
- investigate different answer typing strategies (see Schlobach et al. [19]);
- create (and use) a set of canonical questions to pose to different systems in order to test their knowledge in specific areas, like the University of Bangor in TREC 2002 [10];
- investigate relations among related questions (following [25]).

We note that invoking QA systems as Web services is virtually unlimited – and access to this kind of power (without bothering other people, just using automated systems) allows one to augment knowledge by orders of magnitude, which could then be used intelligently. (In fact, our colleagues from Priberam suggested that a useful byproduct of the QolA pilot was to use its setup to create large collections of validated questions, provided specific interfaces were also developed.)

Another potential impact of QolA is the possible experimentation with the deployment (and use) of special translation web services, that might work differently for specific areas, and for QA in particular, which could be very relevant to crosslingual QA, and CLIR in general. For example, multilingual ontologies and/or gazetteers would come handy for the specific translation of questions, which often have named entities (proper names) which are difficult to deal with by current MT systems. CLEF seems to be the right place to investigate and deploy such specific "query" translation systems, while a web service developed around EuroWordNet [22] would be a sensible candidate for a specific translation provider.

Finally, another interesting experiment would be merging or making sense of information in different languages: if one could use a set of different QA as WS to discover that *London* and *Londres* are the same location (by getting for example the answer in English and in Portuguese about *What is the capital of England?*), or that *Malvinas* and *Falkland* also name a very same location despite completely different names, this might allow systems to answer much more fully questions about these places.

4 Concluding Remarks

We were unfortunately unable to provide the necessary infrastructure for QolA for 2007, and are therefore limited to suggesting the idea to the wider QA community. Some preliminary results are, however, available through `http://www.linguateca.pt/QolA/`: the existence of some QA systems deployed as Web services, the formal WSDL specification of a question answering WS, and general discussion about the whole pilot and how to make it concrete. We still hope QolA will take place some day, not necessarily with us organizing it.

Acknowledgements. The work reported here has been (partially) funded by Fundação para a Ciência e Tecnologia (FCT) through project POSI/PLP/43931/2001, co-financed by POSI, by POSC project POSC/339/1.3/C/NAC, and by SINTEF ICT through project 90512140. We thank Luís Sarmento for providing a translator Web service definition and one baseline, and Sven Hartrumpf, Johannes Levelling, Bernardo Magnini, André Martins and Luís Sarmento for participating in the discussion to make the pilot more concrete. We thank Paulo Rocha and Andreas Limyr for comments on preliminary versions of this paper.

References

1. Ahn, D., Fissaha, S., Jijkoun, V., Müller, K., de Rijke, M., Tjong Kim Sang, E.: Towards a Multi-Stream Question Answering-As-XML-Retrieval Strategy. In: The Fourteenth Text Retrieval Conference (TREC 2005) (2006)
2. Akkiraju, R., Farrell, J., Miller, J., Nagarajan, M., Schmidt, M., Sheth, A., Verma, K.: Web Service Semantics. In: WSDL-S Proceedings of the W3C Workshop on Frameworks for Semantics in Web Services, Innsbruck, Austria, June 9-10, 2005 (2005)

3. Berger, A., Caruana, R., Cohn, D., Freitag, D., Mittal, V.: Bridging the lexical chasm: statistical approaches to answer-finding. In: Proceedings of the 23rd Annual international ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '00), Athens, Greece, July 24 - 28, 2000, pp. 192–199. ACM Press, New York (2000)
4. Brill, E., Lin, J., Banko, M., Dumais, S., Ng, A.: Data-Intensive Question Answering. In: Voorhees, E.M., Harman, D.K. (eds.) Information Technology: The Tenth Text Retrieval Conference (TREC 2001). NIST Special Publication 500-250, pp. 393–400 (2002)
5. Burger, J., et al.: Issues, Tasks and Program Structures to Roadmap Research in Question & Answering (Q&A) (October 24, 2003) http://www.alta.asn.au/events/altss w2003 proc/altss/courses/molla/qa roadmap.pdf
6. Burke, R.D., Hammond, K.J., Kulyukin, V.A., Lytinen, S.L., Tomuro, N., Schoenberg, S.: Question Answering from Frequently Asked Question Files: Experiences with the FAQ Finder System. Technical Report. UMI Order Number: TR-97-05, University of Chicago (1997)
7. Cabral, L., Domingue, J., Motta, E., Payne, T., Hakimpour, F.: Approaches to Semantic Web Services: An Overview and Comparisons. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) ESWS 2004. LNCS, vol. 3053, pp. 225–239. Springer, Heidelberg (2004)
8. Chu-Carroll, J., Prager, J., Welty, C., Czuba, K., Ferrucci, D.: A Multi-Strategy and Multi-Source Approach to Question Answering. In: Proceedings of TREC 2002. NIST Special Publication 500-251, pp. 281–288 (2003)
9. Clarke, C.L.A., Cormack, G.V., Lynam, T.R.: Exploiting redundancy in question answering. In: Proceedings of the 24th Annual international ACM SIGIR Conference on Research and Development in information Retrieval (SIGIR '01), New Orleans, Louisiana, United States, pp. 358–365. ACM Press, New York (2001)
10. Clifton, T., Colquhoun, A., Teahan, W.: Bangor at TREC 2003: Q&A and Genomics Tracks. In: Voorhees, E.M., Buckland, L.P. (eds.) The Twelfth Text Retrieval Conference (TREC 2003). NIST Special Publication: SP 500-255, pp. 600–611 (2004)
11. Dalmas, T., Webber, B.: Answer comparison in automated question answering. Journal of Applied Logic 5(1), 104–120 (2007)
12. Lehnert, W.G.: The process of question answering. Lawrence Erlbaum Associates, Hillsdale, NJ (1978)
13. Magnini, B., Negri, M., Prevete, R., Tanev, H.: Is It the Right Answer? Exploiting Web Redundancy for Answer Validation. In: Proceedings of 40th Anniversary Meeting of the Association for Computational Linguistics (ACL), Philadelphia, USA, July 6-12, 2002, pp. 425–432 (2002)
14. Magnini, B., Giampiccolo, D., Forner, P., Ayache, C., Jijkoun, V., Osenova, P., Peas, A., Rocha, P., Sacaleanu, B., Sutcliffe, R.: Overview of the CLEF 2006, Multilingual Question Answering Track. (In This volume)
15. Maybury, M.T. (ed.).: New directions in question answering. AAAI Press/the MIT Press, Cambridge (2004)
16. O'Reilly, T.: What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software (09/30/2005), http://oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html
17. Santos, D., Cardoso, N.: Portuguese at CLEF. In: Peters, C., Gey, F.C., Gonzalo, J., Müller, H., Jones, G.J.F., Kluck, M., Magnini, B., de Rijke, M., Giampiccolo, D. (eds.) CLEF 2005. LNCS, vol. 4022, pp. 1007–1010. Springer, Heidelberg (2006)

18. Santos, D., Costa, L.: Where is future QA?: One pilot and two main proposals. Presentation at the CLEF 2006 Workshop (Alicante, 22 September 2006) (2006), http://www.linguateca.pt/documentos/SantosCostaQACLEF2006.pdf
19. Schlobach, S., Ahn, D., de Rijke, M., Jijkoun, V.: Data-driven Type Checking in Open Domain Question Answering. Journal of Applied Logic 5(1), 121–143 (2007)
20. Strzalkowski, T., Harabagiu, S. (eds.).: Advances in Open Domain Question Answering. Springer, Heidelberg (2006)
21. Voorhees, E.M., Tice, D.M.: Building a Question Answering Test Collection. In: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Athens, Greece, July 2000, pp. 200–207 (2000)
22. Vossen, P.(ed.).: EuroWordNet: A Multilingual Database with Lexical Semantic Networks. Kluwer Academic Publishers, Dordrecht (1998)
23. W3C Note.: Web Services Description Language (WSDL) 1.1. (2001), http://www.w3.org/TR/wsdl
24. Wilks, Y.A.: Unhappy Bedfellows: The Relationship of AI and IR. In: John, I.T. (ed.) Charting a New Course: Natural Language Processing and Information Retrieval: Essays in Honour of Karen Spärck Jones, pp. 255–282. Springer, Heidelberg (2005)
25. Wu, M., Strzalkowski, T.: Utilizing co-occurrence of answers in question answering. In: Proceedings of the 21st international Conference on Computational Linguistics and the 44th Annual Meeting of the ACL, Annual Meeting of the ACL. Association for Computational Linguistics, Sydney, Australia, July 17 - 18, 2006, pp. 1169–1176. Morristown, NJ (2006)