A Tool Suite for Creating Question Answering Benchmarks

Axel-Cyrille Ngonga Ngomo, Norman Heino, René Speck, Prodromos Malakasiotis
Department of Computer Science, Department of Informatics
University of Leipzig, Athens University of Economics and Business
Augustusplatz 10, 04109 Leipzig, Patision 76, GR10434 Athens
{ngonga|heino|speck}@informatik.uni-leipzig.de, rulller@aueb.gr

Abstract

We introduce the BIOASQ suite, a set of open-source Web tools for the creation, assessment and community-driven improvement of question answering benchmarks. The suite comprises three main tools: (1) the *annotation tool* supports the creation of benchmarks per se. In particular, this tool allows a team of experts to create questions and answers as well as to annotate the latter with documents, document snippets, RDF triples and ontology concepts. While the creation of questions is supported by different views and contextual information pertaining to the same question, the creation of answers is supported by the integration of several search engines and context information to facilitate the retrieval of the said answers as well as their annotation. (2) The *assessment tool* allows comparing several answers to the same question. Therewith, it can be used to assess the inter-annotator agreement as well as to manually evaluate automatically generated answers. (3) The third tool in the suite, the *social network*, aims to ensure the sustainability and iterative improvement of the benchmark by empowering communities of experts to provide insights on the questions in the benchmark. The BIOASQ suite has already been used successfully to create the 311 questions comprised in the BIOASQ question answering benchmark. It has also been evaluated by the experts who used it to create the BIOASQ benchmark.

Keywords: Question Answering, Open-Source Tools, Benchmark Creation

1. Introduction

Assessing the advance of question answering (QA) frameworks requires the provision of manually curated gold standards. Over the last years, several QA benchmark campaigns have produced valuable benchmarks for different QA tasks and made them available to the public. For example, the QALD benchmark (Cimiano et al., 2013) allows evaluating question answering systems for Linked Data. The benchmark data proposed in (Rodrigo et al., 2010) allows measuring amongst others how well a system would perform at university entrance exams. Another benchmark is the BIOASQ benchmark[1], which allows measuring the performance of QA systems on questions created by biomedical experts (Partalas et al., 2013).

However, most of the current QA benchmarks are rather simple in their content as they assume that the question can be answered by using a single type of data, for example textual data, RDF data or data from databases. With the migration of the Web from a document-centred Web to a Semantic Web, QA systems will soon need to address more complex question-answering scenarios where the answer must be gathered from several sources (e.g., documents or fragments of documents, RDF knowledge bases such as DBpedia and ontology repositories such as the OBO Foundry repository) (Nikolov et al., 2013).

Evaluating how well systems perform in these kinds of scenario requires creating complex benchmarks which take several sources of information into consideration. Consequently, furthering the development of better systems requires being able to evaluate the kind of sources on which they perform well. In addition, complex benchmark must allow evaluating the different facets of the question answering process. Given the complexity of such benchmarks, tool support is indispensable to ensure the creation of high-quality benchmarks. Especially, tools for benchmark creation must be able to manage the heterogeneous sources with led to the answer to a question. Moreover, in the case of domain-specific benchmarks (e.g., benchmarks for bio-medicine), the questions have to be created by domain experts which might not be computer experts. Thus, usability is of central importance for benchmark creation tool suites. Finally, we argue that the development of sustainable benchmarks requires taking the input of the corresponding research community into consideration while improving and extending the benchmark. This idea results from previous work on benchmarking, where expert users pointed out flaws in benchmarks that were never taken into account. We address this issue by allowing users to provide their feedback directly to the benchmark questions, which are agents in a social network. By these means, we facilitate the collection of feedback and the refinement of complex benchmarks.

To support the development of realistic benchmarks for complex question answering scenarios, we developed a suite of tools that facilitate the following:

1. Team-based creation of questions via the Web.
2. Search through Web sources or dedicated corpora.
3. Annotation of answers with sources (documents, document fragments, RDF triples, ontological concepts).
4. Assessment of answer quality.
5. Community-driven improvement.

The suite consists of three tools, i.e., the annotation tool BAT, the assessment tool BEAST and the social network BISON. All tools are open-source and available online at http://github.com/AKSW/BioASQ-AT and http://github.com/AKSW/BioASQ-SN. In the following, we first provide a description of each of the tools. Then, we give an overview of a practical use case

[1] http://bioasq.org

within which the tools were employed. Finally, we present evaluation of the tools that was carried out within this use case. Throughout the paper, we will explicate the current instantiation of the tools on the creation of a benchmark for bio-medical question answering. The first version of the benchmark (which was created with exactly these tools) is available at `http://biaosq.lip6.fr`.

2. The Annotation Tool

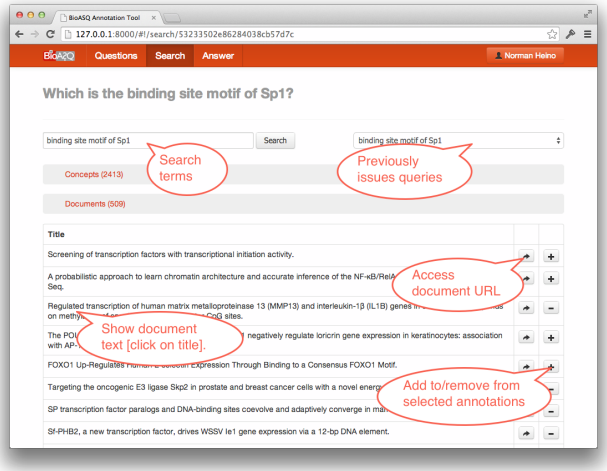

Figure 1: Search and selection window. Users can search for information that can help answer the questions they posed as well as select the relevant results return by several search engines.

The annotation tool BAT was designed to be intuitively usable even for users with for non-experts. While designing the tool, we assumed that the benchmark is created by a team of domain experts that are not necessarily computer-science experts. Thus, we aimed to provide the experts with a simple step-by-step interface in which coherent functionality is bundled into one window. As a result of these considerations, BAT implements a simple benchmark creation paradigm based on the following five steps: authenticate, create, search and select, annotate and store.

The *authentication* ensures that each question created by a certain expert can be assigned to this given expert. Given that only a selected group of individuals are to be able to register and to create a given benchmark, BAT can be configured with a white list of email addresses. Only the owners of these addresses are then able to register and access the benchmark in development. After the authentication, the expert is led to the *question creation* window. Here, he can choose to either work on a question he created in a prior session or to create a new one. The creation of a new question involves deciding upon the question type as well as the question label. Currently, BAT supports boolean, factoid, list and summary questions.

The subsequent *search-and-select* window (see Figure 1) allows search for as well as selecting information necessary to answer the question at hand. This is one of the most tedious steps of the benchmark creation process as the need for a high precision and a high recall requires

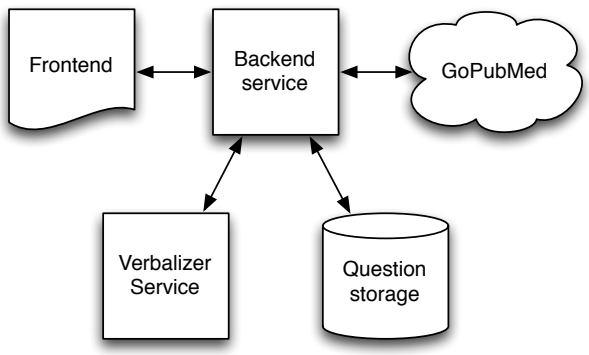

Figure 2: Architecture of the annotation tool

the users to perform manifold queries and read through a large number of answers. To support this task, BAT implements interfaces to search services of which each is connected to one or several data sources. The only condition set upon the services is that they are RESTful and return JSON messages which abide by the message format supported by BAT.[2] Note that the data sources can be of different types. For example, the current deployment of the tool allows accessing unstructured data from PubMed, ontologies such as JoChem and the Gene Ontology as well as RDF data sources from the LinkedLifeData project. A central consideration during the implementation of these services was that the end-users might not be familiar with Web formats for representing structured data. Especially, structured data which were represented by using the Resource Description Framework (RDF) (which is the W3C standard for representing and exchanging structured data) needed to be verbalized. To this end, BAT implements an RDF-to-natural-language (RDF2NL) converter based on SPARQL2NL (Ngonga Ngomo et al., 2013).

The insight behind BAT's RDF2NL converter is that RDF triples (s, p, o) are similar to simple natural-language sentences in English. Given that the subject s of a triple is always a resource, it is sufficient to use its label to verbalize it. The same approach can be used for the object o if it is a resource. Objects that are literals can be verbalized as noun phrases which consists of up to two parts: the literal form of the object and, in the case of units, the data of the object. The more difficult part of the verbalization lies in deciding on whether the predicate is to be verbalized as a noun or as a verb. For example, the predicate `crosses` can be the third person singular form of the verb "to cross" or the plural form of the noun "cross". Here, we rely on the statistical approach presented in (Ngonga Ngomo et al., 2013) to determine how predicates are to be verbalized. By relying on RDF2NL, BAT verbalize the triples shown in the listing below to `Adenosylhomocysteinase is encoded by sahA`.

```
@prefix uniprot:
  <http://purl.uniprot.org/uniprot/> .
@prefix core:
```

[2] See `http://github.com/AKSW/BioASQ-AT` for more information.

```
  <http://purl.uniprot.org/core/> .
@prefix life:
  <http://linkedlifedata.com/resource/> .
@prefix skos:
  <http://www.w3.org/2004/02/skos/core#> .
life:#_4433424E593400E core:encodedBy
  life:#_4433424E59340014 .
life:#_4433424E59340014 skos:prefLabel
  "sahA" .
life:#_4433424E593400E core:fullName
  "Adenosylhomocysteinase" .
```

As shown in Figure 1, BAT also implements provenance tracking functionality. In particular, the interface allows the user to know from which data source the presented information was retrieved. Moreover, BAT also stores the list of queries issued by each user for each question he created.
During the *annotation* step, the answers selected in the previous step are consolidated to one final answer for the question posed. Two steps are necessary to achieve this goal: the formulation of the final answer and the selection of snippets (i.e., text fragments) from text documents previously selected (see Figure 3). This window varies depending on the type of question that was created. For example, the answer to a boolean questions can only be a Yes or a No. Note that in this window, the user can choose to delete some of the previously selected data items for the question. In a final step, the user can chose to finalize a question. This action leads to the question being forwarded to the social network, which is described in section 4. The state of a question can be changed from final to not final at will.

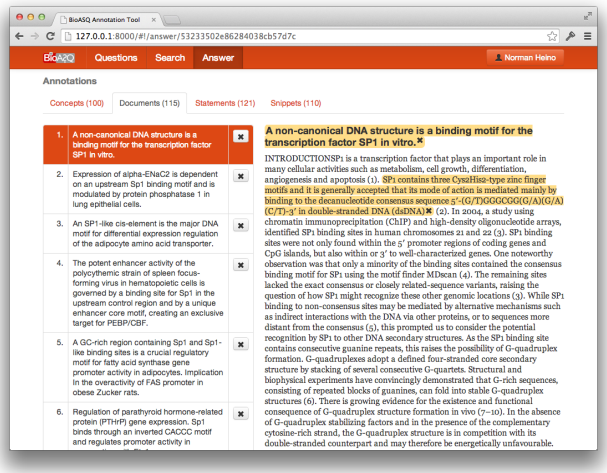

Figure 3: Annotation window. The snippets are marked in yellow. The list of results on the left gives an overview of the selected concepts, documents and statements used as information sources to create the answer. The question is seen on the top right.

3. The Assessment Tool

The assessment tool is built on top of the BAT and reuses most of its functionality. The aim of the assessment tool is twofold. Firstly, it enables the domain experts to evaluate the answers of other parties to the question they created. Secondly, it allows the experts, by reviewing answers of other parties, to revise their own answer. To this end the tool presents a combined view of all answers to a particular question.
In accordance with the BioASQ evaluation framework parts of the answers are presented differently.

- The ideal answers can be scored from 1–5 w.r.t. four dimensions (recall, precision, repetition, readability).
- The exact answer can be modified in light of exact respones of other parties.
- Annotations from other parties can be added to the expert's "golden" set of annotations.

Depending on the source of what is called "other parties" above the BioASQ Assessment Tool can be used for two purposes: the evaluation of system responses and to perform an inter-annotator agreement study. In the first case, the other parties answers are taken from the challenge participants' responses. In the latter case, each question is answered by two different experts and each expert is presented the other expert's response for assessment. The design of the interface is such that the users can always see their gold answers/annotations to questions that they are asked to review (see Figure 4). Moreover, the interface is dynamic and can adapt to different question types. Finally, all information sources that were used by an entity (system, other expert, etc.) to answer the question can also be reviewed. By these means, domain experts can perform an informed assessment.

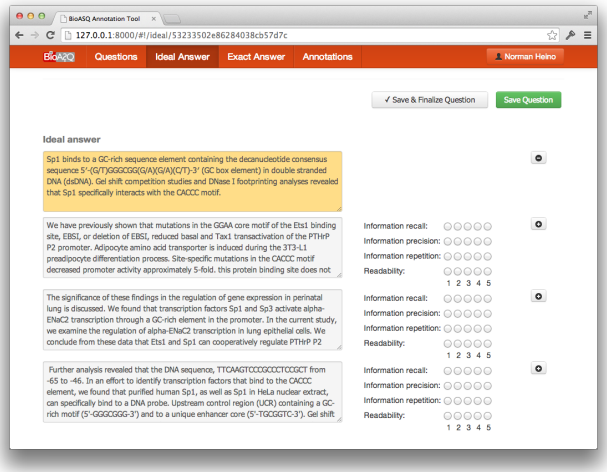

Figure 4: Assessment of ideal system answers. The answer from the gold standard is at the top.

4. The Social Network

Once created, benchmarks are often regarded as monolithic, unchangeable resources. Yet, even manually curated complex benchmarks can contain errors which were not detected by the domain experts while creating the benchmark. We advocate to use a crowd of domain experts to provide insights into possible flaws of benchmarks by deploying a dedicated social network around this benchmark. BISON,

which implements this idea, is an asynchronous social network that allows not only humans but also benchmark entries to be agents. By these means, BISON allows members of the social network to subscribe to (i.e., follow), comment on and refer to questions (see Figure 7 and Figure 8). Based on these commentaries, the experts in charge of the benchmark can iteratively improve their questions to create updated versions of the initial benchmark. Note that the network also implements standard asynchronous network functionality as known from Twitter such as private messaging and following other experts.
BISON has been developed as a modern Web application based on the MEAN stack[3]. This software stack includes the following:

- MongoDB[4], a document-oriented database;
- Express[5], a server-side web framework;
- Angular[6], client-side data-binding and view rendering framework;
- Node[7], a JavaScipt environment for server-side development.

In addition, the Bootstrap[8] CSS framework was used to to achieve a pleasant GUI that is consistent with the Annotation Tool, which also used that framework.

4.1. Architecture

BISON's architectural components can be separated into two subsets: frontend and backend components (see Figure 5). The former reside on the client while the latter are executed on the server. Data exchanged between frontend and backend is comprised entirely of JSON[9] objects.

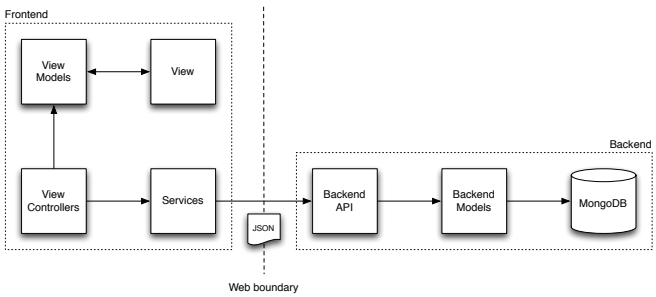

Figure 5: Architecture of the social network. Arrow directions depict control flow. Note the arrow directions between view models and view that denote control flow is in both directions (two-way data binding).

The frontend components include *view controllers*, *services*, *view models*, and *views*. *View controllers* orchestrate interaction of client-side objects. With the help of *services*, which are orthogonal to *view controllers*, they load data from or send data to the backend API. In creating the *view models*, they can transform data if necessary. The Angular rendering system interpolates *views* whenever changes in the associated *view models* are detected and updates the models whenever the *views* changes due to user interaction or other events. Those model changes can also observed in controllers. Alternatively controller functions can be called by the *views* directly.
Due to the backend not containing any views or presentation logic it is kept quite simple. Each request is directed to the respective controller that can use models to interact with the database and retrieve the data desired by the client. Backend controllers can request cookie-based or password authentication. Authentication request are fulfilled before the controller takes action. In case of authentication failure the controller is not executed.

4.2. Data Schema

BISON's data schema consists of three main entities: *Activity*, *Message* and *User*. Both, *Activity* and *Message* are linked to *User* via their *creator* attributes. In addition, *Message* is linked to another user instance via its *to* attribute. Each instance has a *type* attribute that denotes its entity name and an *id* attribute that is allocated by the database.

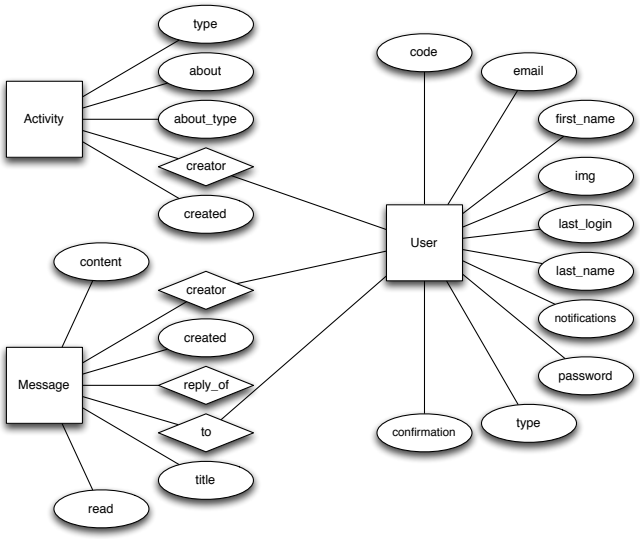

Figure 6: Data schema of the Activity, Message and User entities.

The *Activity* entity subsumes several entities with a similar structure. These are *Follow*, *Comment*, *Question*, and *Vote*. Some of the sub entities need additional properties that are stored directly in the collection. For *Comment* these are *content*, *reply_of*, and *reply_count* while *Vote* stores an additional property called *dir* as vote direction. Additional properties for *Question* are among others *body*, *answer* and *question_type*.
The *Message* entity stores *content* and *title* as well the aforementioned relationships to *User*. In addition, it keeps a self-referencing relation called *reply_ of* in case it is a reply to another message.
The *User* entity stores standard user information like *first_name*, *last_ name*, *email*, *img* and some user prefer-

[3] http://mean.io
[4] http://mongodb.org
[5] http://expressjs.com
[6] http://angularjs.org
[7] http://nodejs.org
[8] http://getbootstrap.com
[9] http://json.org

ences. Authentication related properties like *code*, *confirmation*, as well as an encrypted *password* property are also stored in the *User* entity.

4.3. Functionality

Users must be authenticated by a standard sign in before using BISON, otherwise there is no public information that can be seen. A user's *Home* tab shows a list of activities related to users and questions that the particular user follows. BISON allows users to send each other short messages. The *Messages* tab is where both received and sent messages are displayed. In BISON we tackle discoverability, an important topic in every network, by providing a global view of all activities that takes place in the network. Taking inspiration from Twitter, this view is called *Timeline* and automatically sorted by creation date with the most recently created item being displayed at the top. This way, users are able to discover new questions they might be interested in or people they might want to follow. Each user has an own profile page (see Figure 7) that is partially visible to all other users with some basic information about the user and the users activities, followings and followers.

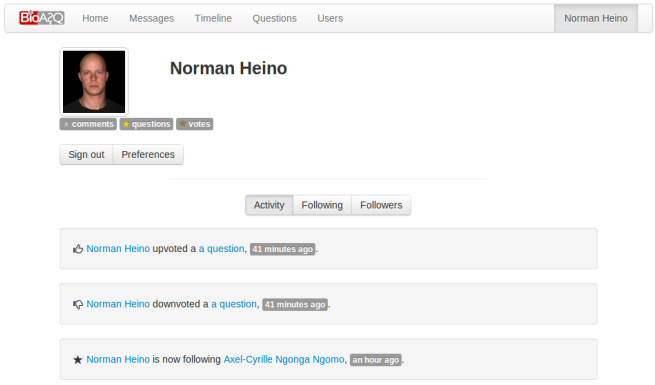

Figure 7: Fragment of a user window.

When visiting the profile page of another user two options are provided: sending a message to the user whose profile page is being displayed or following/unfollowing that user. If the signed in user visits his own profile page, the available options are: sign out or change profile settings. The *Users* tab lists all users currently registered in BISON and can be searched by last name or sorted by several criteria such as first or last name.

Questions are objects of discourse in as well as members of BISON. The *Questions* tab lists all questions currently stored in the network. These can be searched by question terms or sorted by creation date or number of votes. By clicking on the title of a question the answer is revealed along with *concept*, *document*, *snippet*, and *statement* annotations (see Figure 8).

Concept and document annotations are linked to their source documents (e. g. MeSH concept description or PubMed article page). The *Questions* tab empowers users of the network to express their opinion on a particular questions by *voting* in favor of or against a question, *following* a question to be informed about activity that takes places and *commenting* on a question.

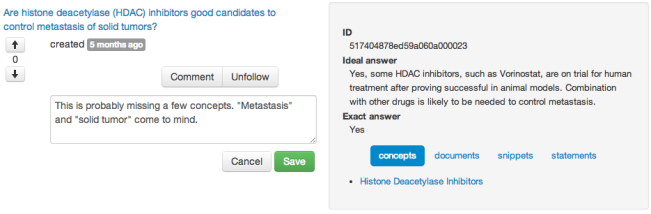

Figure 8: Question window in the social network. This window allows reading questions as well as their annotations, voting for the quality of the questions as well as commenting on the questions.

4.4. Gamification

To improve user engagement, user rewards are assigned for the number of comments, the number of questions and the number of votes each user has done and are displayed below user images (see Figure 7). Each reward comes with three tiers: beginner (bronze), advantaged (silver) and expert (gold).

5. Use Case Study

BAT and BEAST were used by a team of ten bio-medical experts to create and refine the first version of the BIOASQ QA benchmark. This benchmark aims to evaluate how well current question answering tools can answer questions from bio-medical experts such as "`Which are pathological states associated with the formation of DNA G-quadruplexes?`". Four data sources were used to create this benchmark: all PubMed abstracts up to Febrary 2012 (23 million), 800.000 full PubMed documents, 20 RDF knowledge bases (incl. JoCHEM and Diseasome) as well as the MESH ontology. All data sources were indexed using the free search engine Apache Lucene and made available via Apache Solr. Upon a search query, a federated query was sent to all endpoints and returned data from all sources. Relevant data items were selected by the experts and used for later annotation. By these means, a compendium of 311 questions was created.[10] The same team is currently adding new questions to the same benchmark while collecting feedback on the first 311 questions using BISON. In the following, we give an overview of the guidelines provided to the expert for the benchmark creation. The same experts evaluated the tool suite (see section 6.).

5.1. Benchmark Creation Guidelines

Each biomedical expert was asked to formulate questions in English that reflect real-life information needs encountered during his/her work (e.g., in research or diagnosis). Each question was to be stand-alone, i.e., it should not presuppose that any other questions have been asked. For example, it should not contain any pronouns referring to entities mentioned in other questions. For each question, the expert was also expected to provide an answer and other supportive information, as explained below. To formulate each question and to provide the corresponding answer and sup-

[10] The full benchmark is available at `http://bioasq.org/`

portive information, the expert was to follow the following steps.

Step 1: Question formulation. Formulate a stand-alone question in English reflecting real-life information needs. *At least 5 questions of each one of the following four categories* should be formulated by each biomedical expert.

Yes/no (Boolean) questions: These are questions that, strictly speaking, require either a "yes" or a "no" as an answer, though of course in practice a longer answer providing additional information that supports the "yes" or "no" will often be desirable. For example, "*Do CpG islands colocalise with transcription start sites?*" is a yes/no question.

Factoid questions: These are questions that, strictly speaking, require a particular entity (e.g., a disease, drug, or gene) as an answer, though again a longer answer providing additional supportive information may be desirable in practice. For example, "*Which virus is best known as the cause of infectious mononucleosis?*" is a factoid question.

List questions: These are questions that, strictly speaking, require a *list* of entities (e.g., a list of genes) as an answer; again, in practice additional supportive information may be desirable. For example, "*Which are the Raf kinase inhibitors?*" is a list question.

Summary questions: These are questions that do not belong in any of the previous categories and can only be answered by producing a short text summarizing the most prominent relevant information. For example, "*What is the treatment of infectious mononucleosis?*" is a summary question. When formulating summary questions, the experts should aim at questions that they can answer (possibly after consulting the literature) in a satisfactory manner by writing a one-paragraph summary intended to be read by other experts of the same field.

In all four categories of questions, the experts should aim at questions that when converted to PubMedCentral queries, as discussed below, retrieve approximately 10–60 articles (or abstracts). Questions for which there are controversial or no answers in the literature should be avoided.

Step 2: Relevant terms. Form a set of terms that are relevant to the question of Step 1. The set of relevant terms may include terms that are already mentioned in the question, but it may also include synonyms of the question terms, closely related broader and narrower terms etc. For the question "*Do CpG islands colocalise with transcription start sites?*", the set of relevant terms would most probably include the question terms "*CpG Island*" and "*transcription start site*", and possibly also other terms.

Step 3: Information retrieval. Use BAT to formulate a query (Boolean or simple bag of terms) involving the relevant terms of Step 2, as well as to retrieve articles from PubMedCentral that satisfy the query (or abstracts, when only abstracts are available). The query can also be executed against biomedical terminology banks, databases, and knowledge bases, in order to obtain possibly relevant *concepts* (e.g., MESH headings) and relations (e.g., a database may show that a particular disease is known to cause a particular symptom). Relations retrieved from databases and knowledge bases will be shown in the annotation tool as pseudo-natural language statements, hereby called simply *statements*; hence, the experts do not need to be familiar with how information is actually represented in the databases and knowledge bases. Note that when retrieving concepts and statements, advanced search tags are ignored. Furthermore, when retrieving concepts, Boolean operators are also ignored, i.e., Boolean queries are turned into bag of terms queries.

Step 4: Selection of concepts, articles, statements. *All* the concepts of Step 3 that best characterise the question of Step 1 should be selected at this step. Also, *all* the articles of Step 3 that are *possibly relevant* to the question should be selected. By 'possibly relevant' we mean articles that the expert would want to read more carefully in practice, to check if they contain information that is useful to answer the question. At this step, the expert is only expected to skim through the retrieved articles (or their abstracts) to figure out if they are possibly relevant. Finally, *every* statement of Step 3 that provides information that is useful to answer the question should be selected, even if the statement does not provide on its own all of the information that is needed to answer the question. In our example, the following concepts, documents, and statements might be selected:

Step 5: Text snippet extraction. At this stage, the expert should read (or skim through more carefully) the set of possibly relevant articles selected during Step 4. *Every* text snippet (piece of text) that provides information that is useful to answer the question of Step 1 should be extracted, even if the snippet on its own does not provide all of the information that is needed to answer the question. The experts should avoid including in the extracted snippets long pieces of text that do not provide useful information; for example, if only a sentence (or part of a sentence) of a paragraph provides useful information, only that sentence (or part of that sentence) should be extracted as a snippet. On the other hand, the experts should not spend too much time trying to decide exactly where each extracted snippet should start or end; only approximate snippet boundaries are needed. If there are multiple snippets that provide the same (or almost the same) useful information (in the same article or in different articles), *all* of them should be extracted, not just one of them. Snippets can be easily extracted using the annotation tool, much as one might highlight snippets

that provide useful information when reading an article. In our example, the following snippets might be extracted. The numbers in square brackets point to the articles of Step 4 the snippets were extracted from.

Step 6: Query revision. If the expert believes that the snippets and statements gathered during Steps 4 and 5 do not provide enough information to answer the question, the terms of Step 2 and the query of Step 3 should be revised, for example using more or different terms. The process will then continue from Step 3, i.e., the revised query will be used to perform a new search, which may produce different concepts, articles, and statements; the expert will again select (in Step 4) concepts, articles, and statements among those retrieved, and then snippets (in Step 5). BAT provides facilities that allow the concepts, articles, and statements that the expert has already selected (before performing a new search) to be saved, along with the snippets the expert has already extracted. The query can be revised several times, until the expert feels that the gathered information is sufficient to answer the question. If despite revising the query the expert feels that the gathered information is insufficient, or if there seem to be controversial answers, the question should be discarded.

Step 7: Exact answer. At this step, the expert should provide what we call an *exact answer* for the question of Step 1. For a yes/no question, the exact answer should be simply "yes" or "no". For a factoid question, the exact answer should be the name of the entity (e.g., gene, disease) seeked by the question; if the entity has several names, the expert should provide, to the extent possible, all of its names. For a list question, the exact answer should be a list containing the entities seeked by the question; if a member of the list has several names, the expert should provide, to the extent possible, all of the member's names. For a summary question, the exact answer should be left blank. The exact answers of yes/no, factoid, and list questions should be based on the information of the statements and text snippets that the expert has selected and extracted in Steps 4 and 5, respectively, rather than, for example, personal experience.

Step 8: Ideal answer. At this step, the expert should formulate what we call an *ideal answer* for the question of Step 1. The ideal answer should be a one-paragraph text that answers the question of Step 1 in a manner that the expert finds satisfactory. The ideal answer should be written in English, and it should be intended to be read by other experts of the same field. For the example yes/no question "*Do CpG islands colocalise with transcription start sites?*", an ideal answer might be the following:

"Yes. It is generally known that the presence of a CpG island around the TSS is related to the expression pattern of the gene. CGIs (CpG islands) often extend into downstream transcript regions. This provides an explanation for the observation that the exon at the 5' end of the transcript, flanked with the transcription start site, shows a remarkably higher CpG density than the downstream exons."

The ideal answer should be based on the information of the statements and text snippets that the expert has selected and extracted in Steps 4 and 5, respectively, rather than, for example, personal experience. The experts, however, are allowed (and should) rephrase or shorten the statements and snippets, order or combine them etc., in order to make the ideal answer more concise and easier to read etc.

Notice that in the example above, the ideal answer is longer than the exact one ("yes"), and that the ideal answer provides additional information supporting the exact answer. If the expert feels that the exact answer of a yes/no, factoid, or list question is sufficient and no additional information needs to be reported, the ideal answer can be the same as the exact answer. For summary questions, an ideal answer must always be provided.

6. Evaluation

Figure 9: Overall impression for the annotation tool by the team of biomedical experts.

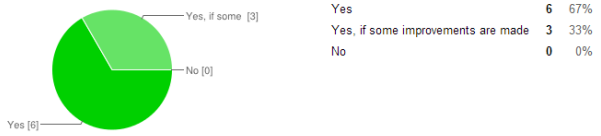

Figure 10: Willingness of the team of biomedical experts to use the annotation tool again.

Figure 11: Willingness of the team of biomedical experts to use the annotation tool for their work (e.g., to organize a search).

We carried out a survey of the annotation and assessment tools with nine of the experts who created the first version of the BioASQ benchmark. The experts were asked 37 questions pertaining to the usability of the tool suite. The

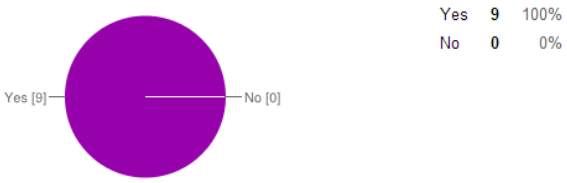

Figure 12: Willingness of the team of biomedical experts to recommend the annotation tool.

answers ranged from 1 to 5 where 1 stood for "Poor" and 5 for "Very good". Figure 9 – Figure 15 show the results of the evaluation. On average, the question creation process was rated with 4.3. This suggests that the users were satisfied with the support provided by the annotation tool during the question creation process. The annotation tool not achieving the perfect score was due to mostly cosmetic requirements required by the end users. These were implemented in the second version of the annotation tool, which is available on GitHub. Interestingly, the experts were even willing to use the tool suite again in the future and even recommend them to others, provided that their requirements were implemented. As this has already been dealt with, we are confident that the evaluation of the second version will be even better. The answer creation process achieved the same average score as the question creation process. The most difficult part of the answer creation, i.e., the snippet annotation, achieved an average score of 3.83. This is due to minor bugs in the first version of the tool that have already been corrected. The assessment tool achieved similar results.

Figure 13: Overall impression for the assessment tool by the team of biomedical experts.

Figure 14: Willingness of the team of biomedical experts to use the assessment tool again.

7. Conclusion and Future Work

We presented the BioASQ tool suite for creating question answering benchmarks. The rationale behind the tool suite

Figure 15: Willingness of the team of biomedical experts to recommend the assessment tool.

is to provide an ecosystem of tools for the creation and iterative improvement of question answering benchmarks. The tools are all open-source and can be extended and reused at will. They were implemented using state-of-the-art Web technology to ensure scalability, easy deployment and use. Our use case and evaluation suggest that the tools can be used intuitively even by domain experts with no special affinity to computer science. Preliminary results on the now updated versions of the tools suggest that the bio-medical experts who are currently creating the second version of the BioASQ benchmark find the tool easier to use and are more efficient in creating questions. An evaluation of the new version of the tools will be carried out in the near future.

Acknowledgment

The authors of this paper were supported by the FP7 BioASQ project with the Grant Agreement No. 318652.

8. References

Cimiano, P., Lopez, V., Unger, C., Cabrio, E., Ngonga Ngomo, A.-C., and Walter, S. (2013). Multilingual question answering over linked data (qald-3): Lab overview. In *CLEF*, pages 321–332.

Ngonga Ngomo, A.-C., Bühmann, L., Unger, C., Lehmann, J., and Gerber., D. (2013). Sorry, i don't speak sparql — translating sparql queries into natural language. In *Proceedings of WWW*.

Nikolov, A., Schwarte, A., and Hütter, C. (2013). Fedsearch: Efficiently combining structured queries and full-text search in a sparql federation. In *International Semantic Web Conference (1)*, pages 427–443.

Partalas, I., Gaussier, É., and Ngonga Ngomo, A.-C. (2013). Results of the first bioasq workshop. In *Proceedings of the BioASQ Workshop*.

Rodrigo, Á., Peñas, A., Hovy, E. H., and Pianta, E. (2010). Question answering for machine reading evaluation. In *CLEF (Notebook Papers/LABs/Workshops)*. Citeseer.