Procesamiento del Lenguaje Natural, Revista nº 60, marzo de 2018, pp. 21-28 recibido 03-11-2017 revisado 08-01-2018 aceptado 14-02-2018

Not all the questions are (equally) difficult. An hybrid approach to CQA in Arabic

No todas las preguntas son (igualmente) difíciles, una aproximación híbrida a la CQA en árabe

Imane Lahbari[1], **Horacio Rodríguez**[2], **Said Ouatik El Alaoui**[1]
[1]Laboratory of Informatics and Modeling, Faculty of Sciences Dhar El Mahraz, Sidi Mohamed Ben Abdellah University, Fez, Morocco
[2]Dep. of Computer Science, Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
imane.lahbari@usmba.ac.ma, horacio@cs.upc.edu, s_ouatik@yahoo.com

Abstract: In the past we faced the problem of Community Question Answering using an unified approach. Some of the questions, however, are easier to be approached by a conventional rule-based system. In this paper we explore this direction.
Keywords: Community question answering, arabic processing

Resumen: En el pasado hemos abordado la búsqueda de respuestas en comunidades usando un enfoque uniforme. Sin embargo, algunas preguntas pueden ser respondidas utilizando métodos basados en reglas. En este tratajo exploramos esta dirección.
Palabras clave: Búsqueda de respuestas, procesamiento del árabe

1 Introduction

Community Question Answering (*CQA*) has become increasingly popular in the last years. It is seen as an alternative to both classical Information Retrieval and more specific Question Answering (*QA*) tasks. Both general purpose, as Yahoo!Answers (*Y!A*)[1], and topic-specific communities, such as StackOverflow (*SO*)[2], have got an impressive growth.

CQA purpose is to provide to users pertinent answers to their questions by identifying, among a set (sometimes a thread) of question answer pairs, questions that are similar to the original one.

The SemEval Task 3 subtask D (Nakov et al., 2017) asks, given a question in Arabic, denoted the original question, and a set of the first 9 related questions (retrieved by a search engine), each associated with one correct answer, to re-rank the 9 question-answer pairs according to their relevance with respect to the original question. Figure 1 presents a fragment of a query thread containing an Arabic query (a), and its English translation, carried out using Google Translate API (b). It is worth noting from this example: i) the high density of medical terms as seen in (c), ii) the occurrence of English terms embedded within the Arabic texts, that could complicate the linguistic process of Arabic texts, iii) the relatively high overlapping of terms between the query texts and the texts of query/answer pairs in both Arabic texts and English translations, and iv) The relatively low quality of English translations, that could result on low accuracy of the linguistic process of English texts.

We developed in the past a CQA system based on the combination of a number of atomic classifiers, which was evaluated in the framework of SemEval 2017 Task 3 subtask D, getting good results. Some of the questions, however, seem to be easier to be approached by a conventional rule-based system. In this paper we explore this direction.

2 Related works

QA, i.e. querying a computer using Natural Language, is an old objective of Natural Language Processing. Though initially *QA* systems focused on factual questions (who, where, when, Y/N, etc.), increasingly, the scope of *QA* has become wider, facing complex questions, list questions, definitional, why questions, etc. In parallel the *QA* sys-

[1]http://answers.yahoo.com/
[2]http://stackoverflow.com/

ISSN 1135-5948. DOI 10.26342/2018-60-2

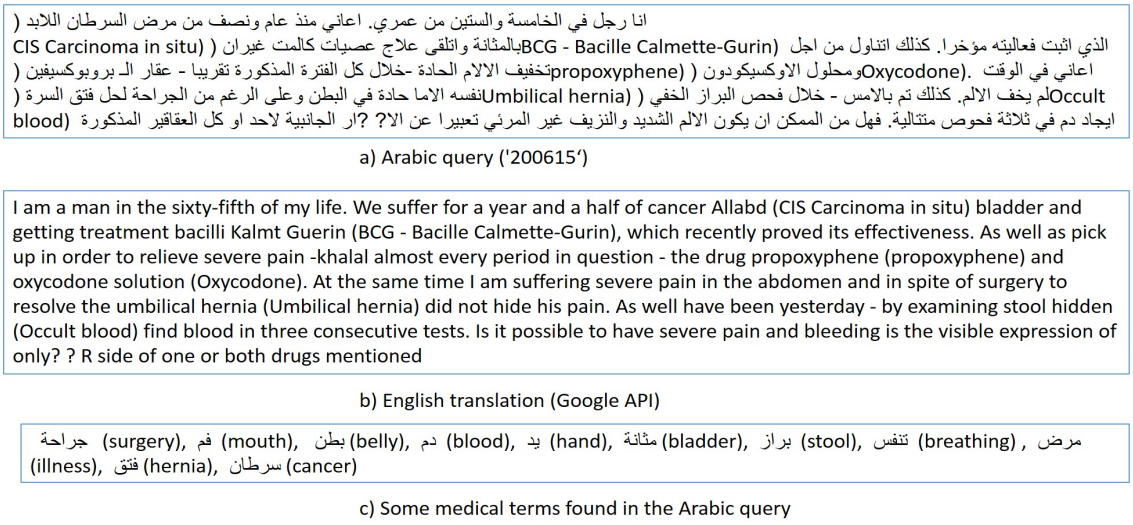

Figure 1: Query thread fragment

tems have suffered a process of specialization: domain-restricted *QA*, *QA* for comprehension reading, *QA* over Linked Data, or *CQA*.

CQA differs from conventional *QA* systems basically on three aspects:

- The source of the possible answers, that are threads of queries and answers activated from the original query. So, the document or passage retrieval components, needed in other *QA* systems can be avoided or highly simplified;
- The structure of the threads and the available metadata can be exploited for the task
- The types of questions include the frequent use of complex questions, as definitional, why, consequences, how_to_proceed, etc.

Many approaches have been applied to the task (Nakov et al., 2016; Nakov et al., 2017; El Adlouni et al., 2016; El Adlouni et al., 2017) for details and references.

Most of the systems use, as core features or combined with others, textual features, superficial (string-based), syntactic, and, less frequently, lexico-semantic (knowledge-based), usually reduced to similarity or relatedness measures between the textual components of the thread (query, query/answer pairs), Gomaa and Fahmy (2013) present an excellent survey of these class of features;

Most of the research on QA has been applied to English language. There are, however, interesting examples in other languages, including Arabic. Two of the most useful references for Arabic *QA* are the thesis of Yassine Benajiba (Benajiba, 2009) and Lahsen Abouenour (Abouenour, 2014). Focusing on rule-based approaches, interesting systems are: QARAB, (Hammo, Abu-Salem, and Lytinen, 2002), for Factoid questions, DefArabicQA, (Trigui, Belguith, and Rosso, 2010), for Definitional questions, and, EWAQ, (Al-Khawaldeh, 2015), an Entailment-based system.

3 Our SemEval 2017 system

In this section we present our previous system, (El Adlouni et al., 2017; El Adlouni et al., 2016), evaluated in the framework of SemEval-2017 Task 3 D, on which we will further include our rule-based module described in this article. Our official results in his contest were rather good, second (but from only 3 teams) in *MAP* and first in *accuracy*. Our system combined different basic classifiers in several ways.

The overall architecture of our system is shown in Figure 2. As can be seen, the system performs in four steps:

- A preparation step, aiming to collect domain (medical) specific resources;
- A learning step, for getting the models of the classifiers;

- A classification step, for applying them to the test dataset. These two steps are applied independently for each of the basic classifiers;
- A last step combines the results of the atomic classifiers for obtaining the final results.

We describe each step next.

3.1 Overall description

A core component of our approach is the use of a medical terminology, covering both Arabic and English terms and organized into three categories: *body parts*, *drugs*, and *diseases*. We decided to use this resource taking into account the origin of the datasets for task D: *medical fora*. The terminology was automatically collected as reported in El Adlouni et al. (2017). The process of collecting it was performed in a multilingual setting (7 languages were involved). Some of the languages provided available medical resources (as SnomedCT for English, French, and Spanish, DrugBank, and BioPortal for English, and other), while translingual links were obtained from DBPedia (English, French, German, and Spanish) through the use of *same_as* and *label* properties (Cotik, Rodriguez, and Vivaldi, 2017). The final figures for Arabic and English can be found in Table 1.

After downloading the training (resp. test) Arabic dataset we translated into English all the Arabic query texts and all the Arabic texts corresponding to each of the query/answers pairs. For doing so we have used the Google Translate API[3].

For processing the English texts we have used the Stanford CoreNLP toolbox[4] (Manning et al., 2014). For Arabic we have used *Madamira*[5] (Pasha et al., 2015).

The results obtained were then enriched with WordNet synsets both for Arabic (Rodríguez et al., 2008) and English (Fellbaum, 1998). Also the sentences extracted were enriched with Named Entities corresponding to the medical terminologies collected in the preparation step[6].

[3]translate.google.com

[4]http://stanfordnlp.github.io/CoreNLP/

[5]http://nlp.ldeo.columbia.edu/madamira/

[6]Some of these terms are classified as ORG or MISC, by the linguistic processors, others are simply not recognized as Named Entities.

Medical Category	English	Arabic
Body Part (BP)	25,607	1,735
DISEASE	292,815	3,352
DRUG	87,254	2,149

Table 1: Medical terms datasets

No WSD was attempted. As detection of medical multiword terms is poor in StanfordCore and Madamira, a post process for locating them when occurring in our medical terminologies or WordNets was carried out.

After that, a process of feature extraction was performed. This process is different for each atomic classifier and will be described in next sections. Finally, a process of learning (resp. classification) is performed. Also these processes differ depending on the involved classifier and will be described next.

Our approach for learning consists on obtaining a set of N classifiers. Besides classifying, a score or credibility value is provided by the classifier that can be used in fact as a ranker[7].

3.2 Atomic classifiers

The set of atomic classifiers was selected in order to deal with the different aspects that seem relevant and have been successfully applied to similar tasks: textual features, IR, topics, dimensionality reduction, etc.). The atomic classifiers used by our system are the following:

- Basic lexical string-based classifiers, i.e. *Basic_ar* and *Basic_en*, see details in El Adlouni et al. (2017);
- A simple *IR* system, using *Lucene* engine;
- A *LSI* system, learned from different datasets;
- A topic-based *LDA* system;
- A *Embedding* system.

3.2.1 Basic classifiers

We use two equivalent basic atomic classifiers, one applied to Arabic (*basic_Ar*) and the other to English (*basic_En*). The basic classifiers use three sets of features: shallow linguistic features, vectorial features, and domain-based features. As shallow linguistic features we use most of the 147 features proposed in Felice, M. (2012).

[7]Because the task we face consists on both classifying and ranking.

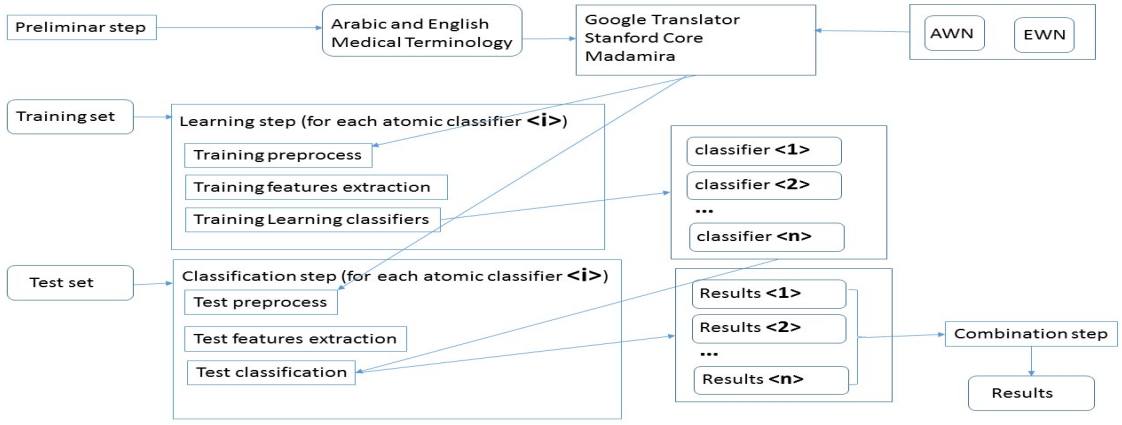

Figure 2: Train and testing pipelines

It is worth noting the importance of the medical features: While only 57 out of 147 basic features were used by the classifier, the whole set of medical features (16) were used. Ranking the features by decreasing accuracy 4 medical features (i.e. 25% of them) occur between the 20 highest ranked features.

We have used for learning the *Logistic Regression* classifier in the Weka toolkit[8].

3.2.2 Lucene classifier

Using Lucene[9], we index the pairs by using all possible combinations (q, q^i, a^i, and $q^i \oplus a^i$) searching thereafter for each pair $<q, q^i>$ for obtaining a set of hits or document with their respective relevance to the query.

3.2.3 LSI and Embedding classifiers

For dealing with dimensionality reduction we have used two techniques, LSI and embeddings. LSI was used to have dense representations of our sentences by using SVD. Various corpora were used for that matter including Wikipedia latest dump (January 2017), Webteb.com, altibbi.com and dailymedicalinfo.com which are specialized Arabic websites for medical domain articles. For embeddings we used the *doc2vec* approach described in Le and Mikolov (2014).

3.2.4 LDA classifier

As for LSI, LDA is used to produce dense representations for our sentences using the implementation included within *Gensim* (Hoffman, Bach, and Blei, 2010). Our aim is to capture topics implicitly occurring within the questions.

[8]http://www.cs.waikato.ac.nz/ml/weka/
[9]https://lucene.apache.org/

3.3 Combinations

Our combiner receives as input a set of atomic result and scores and returns an overall result and score.

The combiner is driven by a set of hiperparameters:

- *scoring form*, i.e. 'max' or 'ave';
- *thresholding form*, i.e. None, 'global' or 'local';
- *thresholding level*, i-e. 0.2, 0.4, 0.6, 0.8;
- *result form*, i.e. 'max', 'voting', 'coincidence'.

We have used grid search for setting the best combination of the hiper-parameters, using the development dataset.

4 Experimental framework

The scorer made available by SemEval organizers provides a range of evaluation metrics to assess the quality of the proposed model, the two most important are *MAP* and *Accuracy*. The former which stands for the Mean Average Precision is the official score of the competition and is based on the top ranked question-answer pairs for each original question leveraging the value computed for our score in our dataset on a scale from 0 to 1. The latter is based on the binary result (relevant or not).

4.1 Results

In Table 2 a summary of the Official results of Semeval 2017 Task 3 Subtask D, are presented (all but last row).

Team	MAP	Acc
GW_QA-primary	0.6116	0.6077
UPC-USMBA-primary	0.5773	0.6624
QU_BIGIR-primary	0.5670	0.4964
UPC-USMBA-with rules	0.5786	0.6747

Table 2: Official results of the task

ruleset	accuracy: m	a	overall
Arabic	0.757	0.559	0.635
English	0.763	0.549	0.652
union	0.755	0.54	0.629
intersection	0.921	0.842	0.875

Table 3: Accuracy of rule-based on test set

Regarding *MAP*, and so looking at the official rank, we were placed in the middle (2nd from 3 participants). Regarding *accuracy* we are placed on the top of the rank. We analyzed the results in the test dataset of our atomic classifiers (with different parameterization) and combinations. The *MAP* for the atomic classifiers (using the best parameters got in training) range from 55 to 58.32. All the atomic results were outperformed by our combiner run but *Lucene*, which obtained our best result, 58.32.

5 Including a rule-based model

A careful examination of errors in our previous approach revealed that some apparently easy cases, as those shown in Table 4, failed to be correctly classified. We saw that some of the original queries, though not the majority, corresponded to factoid questions and could be approached by a conventional rule-based system. So, we developed a rule-based model for facing factoid questions, i.e. cases where a clear, although possibly not unique, objective can be extracted from the text. This rule-based model will be later included into our combination approach.

Consider, for instance, a question beginning with "What is the cause of", and containing close to it a disease name. This question can be intuitively classified as *CAUSE_DISEASE* and parameterized with the tag *DISEASE* with the extracted name as value.

Our rule-based approach consists of the following steps:

- We build a set of question types (QT), *QTS*. QT are domain-restricted semantically-driven tags. *QTS* consists of 27 QT, including DEFINITION_DISEASE, CAUSE_DISEASE, SIDE_EFFECTS_DRUG, etc. The later can be paraphrased as "given an instance of a DRUG, what are its possible side effects (clinical findings)?";
- For each QT we have build 4 sets of classification rules, for Arabic and English, manually and automatically built. For building the rules the training material of SemEval was used. The process of building these rulesets is detailed in section 5.1. The process resulted in 27 Manual Arabic rules, 29 Automatic Arabic rules, 52 Manual English rules, and 83 Automatic English rules. An average of 8 rules per QT have been built;
- Extraction rules are straightforward and language independent. We have manually built one for each QT;
- We have built a rule-based classifier that can be applied to the original question q or and to any of the questions in the pairs of the thread (q^i). The same classifier is used for both languages using the corresponding rule-set. The classifier returns for each case zero (in most of the cases) or more QT from *QTS*. We have built, too, a rule-based extractor that can be applied to the answers in the pairs of the thread (a^i).

I suffer from psoriasis since a long time I want ...
Is there a cure for psoriasis in Jordan ?
Is there a cure for psoriasis ?
Do Hnal cure for psoriasis
I have psoriasis in the top of the ...

Table 4: Some questions in the thread of "What the treatment of psoriasis ?"

The application of the classifiers/extractors is as follows: The process of *classification rules* consist of obtaining the QT, deriving from it the *Expected Answer Type* (EAT), and set the *Mandatory Constraints* (MC) and *Optional Constraints* (OC)[10]. For example, for the case of QT SIDE_EFFECTS_DRUG, the MC is reduced to the tag 'DRUG' associated with the specific name quoted in the question.

[10] Although both MC and OC are generated, only the former are considered in this paper.

After applying the classifiers to all the cases a pair $<q,\ q^i>$ is considered relevant when:

- q and q^i are classified into the same QT (not necessarily by the same rule or language);
- The involved MC are compatible;
- An extraction rule can be applied to a^i using the same MC.

The sets of rules have been evaluated in terms of accuracy over the test set. The results are shown in Table 3. We depict the accuracy of the Arabic and English rulesets, their union and their intersection for manual, automatic, and overall rulesets. It is worth noting the serious degradation of accuracy from manual to automatic rules and the relative similarity of performance for Arabic and English.

5.1 Building the rulesets

Classification rules perform on all the questions, both q and q^i.

A rule consists of a sequence of conditions followed by a sequence of actions (usually only an action is included into the rule). Actions are executed only when all the conditions are satisfied. Each condition (and action) returns a value that can be used by the following ones. The action part of the rule is in charge of building the constraints that will be evaluated by the extraction rules. Extraction rules are associated to the MC and OC obtained by classification rules, Usually they are reduced to check whether the entities (diseases, drugs, body parts) contained in MC occur on the answer text. There are basically three kind of conditions in classification rules (see some examples just below):

- Those checking for the occurrence of textual patterns referring to words, lemmas, pos, NEs, ... on the text of the question;
- Those looking for the occurrence of medical entities (DISEASE, DRUG, BP) from our medical vocabularies;
- Those establishing distance constraints between the tokens located in 1 and 2.

A total of 22 condition predicates have been built to be used within the rules. In average each rule contains 5 conditions. Some of these predicates are the following:

```
def CQARule_SYMPTOMS_DISEASE_en_2(lang,qT):
# 20006, 100739, What are the symptoms of bird-pig disease ...
pattern = [u'l(what)', 'sk(0:2)', u'l(symptom)', 'sk(0:2)', 'n(DISEASE)', 'sk(*)']
id = 'CQARule_SYMPTOMS_DISEASE_en_2'
p = QtclassRegularPattern(id, pattern)
p.prepare()
r = QTclassrule('SYMPTOMS_DISEASE_en_2',qT,lang)
r.addCondition(QTclasscondition("c0","thereAreTriggers(l, qT, s)"))
r.addCondition(QTclasscondition("c0","noStigmas(l, qT, s)"))
r.addCondition(QTclasscondition("c0","noYNQuestion(l,s)"))
r.addCondition(QTclasscondition(
   "c1","applyComplexPattern('CQARule_SYMPTOMS_DISEASE_en_2',s)"))
r.addAction(QTclassAction("a1",'addInvolvedToMandatory_1([("c1",-1)],[])'))
return r
```

Figure 3: Example of rule

- thereAreTriggers: Checks whether the question contains at least one of the triggers of the QT, i.e. terms heavily pointing to this QT.
- noStigmas: Checks whether the question contains stigma terms, i.e. terms forbidden for the QT, usually triggers of the other QT;
- noYNQuestion: Checks whether a pattern for a YN question occurs;
- applySimplePattern: Checks whether the regular expression in *pattern* is satisfied by the question;
- applySortedPatterns: Sorts the list of strings in *patterns* by decreasing length and checks their occurrence in the question;
- existInInstancesInOntology: Checks whether instances of the elements of *involved* occur in the question;
- checkDistanceConstraint: Checks the distance constraints, contained in *constrains* between the tokens located in previous conditions.

We tried to build rules for the most used patterns. Within the training data set, people use to ask about their own issues. We studied this data set and we extract the most used expressions. In general, people ask about diseases, drug or body parts (BP) which are automatically detected by our system. The interrogative patterns, *IP*, are the first component for building any rule, then we describe the whole expression. For each expression, we define a few tokens after the *IP*, then we add the extracted diseases (or drug, or BP).

An example of Python function for building a manual English rule is shown in Figure 3. The function for creating

the rule has two parameters, the language and the *QT*, "English" and "SYMPTOMS_DISEASE" in this case. The identification of the rule is defined as id = "CQARule_SYMPTOMS_DISEASE_en_2". The rule includes as a comment an example of application: "What are the symptoms of bird-pig disease ..." The rule owns an internal parameter, *pattern*, that can be paraphrased as: The question starts with a token whose lemma should be "what", next zero to two tokens could be skipped, the next token has be have a lemma "symptom", new skipping of up to two tokens and a token corresponding to a NE of type "DISEASE". Finally the rest of tokens could be skipped.

This rule contains 4 conditions and 1 action. The first three conditions apply "thereAreTriggers", "noStigmas", and "noYNQuestion". The results of all these three conditions are assigned to the variable "c0", not later used. The fourth condition checks whether the complex pattern is satisfied. The list of tokens, "what", "symptom", and the disease, is assigned to variable "c1". The only action simply builds the *MC* including the last member of "c1", i.e. the name of the disease.

Building manually the set of classification rules resulted on 27 rules for Arabic and 52 for English. Although these rules offered a nice precision, the recall was very low. So, we decided to complement these rulesets by means of a semi-automatic procedure involving a very low human intervention. This process is as follows:

For each *QT* and for both languages, all the manual rules were applied to all the questions (q and q^i) in the training set. We collected all the cases of success. We obtained in this way a set of question texts (444 for Arabic and 746 for English). For each of these texts we collected the occurring n-grams (up to 5-grams including up to 2 skips). Using a tf*idf weighting, the most frequent n-grams were obtained. This resulted on 3,958 n-grams for Arabic and 1,702 for English. From this information we built two matrices of 27 rows corresponding to *QTS* and 3,958 (1,702) columns, number of selected n-grams. This matrices were manually revised and some of the columns were removed. Then for each row the involved medical entities (DISEASE, DRUG, BP, ANY) and their distance constraints were manually added. After this process the set of automatic rules is easily generated.

In Table 3 global accuracy of the set of rules obtained on the test set are presented.

The rule-based classifier has been incorporated to our combiner getting the result shown in the last row of Table 2. Both MAP and accuracy got an improvement though only the later is significant.

6 Conclusions and future work

Our official results on the contest have been rather good, second (but from only 3 teams) in *MAP* and first in *accuracy*. The inclusion of our rule-based classifier has consistently outperformed *accuracy*. *MAP* has also improved but the improvement is not significant. This is due to the fact that a very limited number of cases has changed, so, although the binary results (classification) have improve, the scores (ranks) have changed only marginally.

Our next steps will be:

- Performing an in depth analysis of the performance of our two rulesets, analyzing the accuracy of each rule and cross comparing the rules fired in each language. It is likely that if a rule has been correctly applied to a pair for a language a corresponding rule in the other language should be applied as well, so modifying an existing rule or including a new one could be possible. This line of research can be followed for both manual and automatic approaches.
- Using a final ranker (not a simple classifier) over the results of our atomic classifiers for trying to improve our *MAP*.
- Trying others NN models as CNN and LSTM that have provided good results for English.
- Extending the coverage of our medical terminologies to other medical entities (procedures, symptoms, clinical signs and findings).

Acknowledgments

We are grateful for the suggestions from three anonymous reviewers. Dr. Rodríguez has been partially funded by Spanish project "GraphMed" (TIN2016-77820-C3-3R).

References

Abouenour, L. 2014. *Three-levels Approach for Arabic Question Answering Systems.* Ph.D. thesis, University Mohammed V Agdal, R, (Morocco).

Al-Khawaldeh, F. T. 2015. Answer extraction for why arabic question answering systems: Ewaq. In *World of Computer Science and Information Technology Journal (WCSIT)*, pages 82–86.

Benajiba, Y. 2009. *Arabic Named Entity Recognition.* Ph.D. thesis, UPV Valencia (Spain).

Cotik, V., H. Rodriguez, and J. Vivaldi. 2017. Arabic medical entity tagging using distant learning in a Multilingual Framework. In *Journal of King Saud University-Computer and Information Sciences, vol. 29*, pages 204–211.

El Adlouni, Y., I. Lahbari, H. Rodríguez, M. Meknassi, S. O. El Alaoui, and N. Ennahnahi. 2016. Using domain knowledge and bilingual resources for addressing community question answering for arabic. In *4th IEEE International Colloquium on Information Science and Technology, CiSt 2016, Tangier, Morocco, October 24-26, 2016*, pages 368–373.

El Adlouni, Y., I. Lahbari, H. Rodríguez, M. Meknassi, S. O. El Alaoui, and N. Ennahnahi. 2017. Upc-usmba at semeval-2017 task 3: Combining multiple approaches for cqa for arabic. In *Proceedings of SemEval 2017.*

Felice, M. 2012. Linguistic Indicators for Quality Estimation of Machine Translations. In *Master's thesis. University of Wolverhampton, UK.*

Fellbaum, C. 1998. WordNet: An Electronic Lexical Database. In *MIT Press, Cambridge, Mass (USA).*

Gomaa, G. and A. Fahmy. 2013. A Survey of Text Similarity Approaches. In *International Journal of Computer Applications 04/2013; 68(13).*

Hammo, B., H. Abu-Salem, and S. Lytinen. 2002. Qarab: A question answering system to support the arabic language. In *Workshop on Computational Approaches to Semitic Languages*, pages 1–11.

Hoffman, M., F. R. Bach, and D. M. Blei. 2010. Online learning for latent dirichlet allocation. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23.* Curran Associates, Inc., pages 856–864.

Le, Q. and T. Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1188–1196.

Manning, C., M. Surdeanu, B. J., J. Finkel, S. Bethard, and D. McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *ACL System Demonstrations*, pages 55–60.

Nakov, P., L. Màrquez, A. Moschitti, W. Magdy, H. Mubarak, A. A. Freihat, J. Glass, and B. Randeree. 2016. Semeval-2016 task 3: Cqa. In *Proceedings of SemEval '16*, San Diego, California. ACL.

Nakov, P., D. Hoogeveen, L. Màrquez, A. Moschitti, H. Mubarak, T. Baldwin, and K. Verspoor. 2017. SemEval-2017 task 3: Community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation*, SemEval '17, Vancouver, Canada. ACL.

Pasha, A., M. Al-Badrashiny, M. Diab, N. Habash, M. Pooleery, O. Rambow, and R. Roth. 2015. Madamira 2.1. In *Center for Computational Learning Systems Columbia University*, pages 55–60.

Rodríguez, H., D. Farwell, J. Farreres, M. Bertran, M. Alkhalifa, M. A. Martí, S. Elkateb, W. Black, J. Kirk, A. Pease, P. Vossen, and C. Felbaum. 2008. Arabic WordNet: Current state and future extensions. In *Proceedings of the Fourth Global WordNet Conference, Szeged, Hungary*, pages 387–405.

Trigui, O., L. H. Belguith, and P. Rosso. 2010. DefArabicQA: Arabic Definition Question Answering System. In *7th Workshop on Language Resources and Human Language Technologies for Semitic Languages*, pages 40–45.