ISSN 2007-9737

Question Answering Passage Retrieval and Re-ranking Using N-grams and SVM

Nouha Othman[1], Rim Faiz[2]

[1] Université de Tunis, Institut Supérieur de Gestion de Tunis, LARODEC, Tunisia

[2] Université de Carthage, IHEC Carthage, LARODEC, Tunisia

othmannouha@gmail.com, rim.faiz@ihec.rnu.tn

Abstract. Over the last few decades, with the meteoric rise of Information Technology, Question Answering (QA) has attracted more attention and has been extremely explored. Indeed, several QA systems are based on a passage retrieval engine which aims to deliver a set of passages that are most likely to contain a relevant response to a question stated in natural language. In an attempt to enhance the performance of existing QASs by increasing the number of generated correct answers and ensure their relevance, we propose a novel approach for retrieving and re-ranking passages based on n-grams and SVM models. The core principle is to first rely on the dependency degree of n-gram words of the query in the passage to retrieve correct passages. Then, an SVM based model is used to improve passage ranking incorporating various lexical, syntactic and semantic similarity measures. Emperical evaluation performed with the CLEF dataset demonstrates the merits of our approach: the results obtained by our implemented system transcend that of other previously proposed ones.

Keywords. Question answering, passage retrieval, passage-ranking, n-grams, SVM, similarity measures.

1 Introduction

Question Answering is a specialized area in the field of Information Retrieval that ought to directly return a short and precise answer to a human natural language question rather than a full list of documents and web pages. It is deemed to be a longstanding problem that has been widely studied over the last few years [26], fuelled by the QA tracks in the yearly campaigns TREC[1] and CLEF[2]. Today, the amount of data and information is steadily rising owing to the blossom of Information Technology. Therefore, the need to direct information is more than ever necessary and has imposed QA as a crucial problem for advancing Information Retrieval, Natural language Processing [14] and web search [13]. It is worth noting that there are two main categories of QASs: Open-domain QA deals with questions about everything like [1], and closed-domain QA deals with questions under a specific domain such as [26]. Note that in this paper, we deal with open domain QA since the techniques applied are not tailored toward a precise domain.

Generally, QA systems (QAS)s are composed of three core modules, namely question classification, information retrieval, and answer extraction, each of which constitutes a challenging sub-problem still open for further research endeavor [2]. The goal of the first module is to process the question given in natural language in order to represent it in a simple form but with more information such as the question class, the answer type, the focus and keywords. Information retrieval module attempts to reduce the search space for finding an answer by finding from the collection of documents the top N paragraphs that match the output of question analysis. Answer selection aims to select relevant answers from the

[1] http://trec.nist.gov/
[2] http://www.clef-initiative.eu/

relevant paragraphs retrieved by the text retrieval module based on the result of question analysis. In particular, information retrieval module mostly involves a passage retrieval (PR) engine that is advocated as the key component of a typical QAS, since its performance significantly affects that of the whole system [29]. PR attempts to reduce the search space by shortening the large-scale documents into short passages in order to make faster the system. We emphasize that the response time of a QAS is very crucial owing to the interactive nature of QA. Undoubtedly, a correct answer to a posted question can be found only when it already exists in one of the retrieved passages.

Indeed, extracting a relevant passage to a given natural language question over a huge document collection remains a challenging task that has been extensively researched over many years. Unfortunately, major existing QASs have not yet been able to directly deliver from a vast repository a correct answer to a given question even though It is contained within the repository. Hence, we suppose that, we could increase the number of relevant answers to a posted question if we improve the PR engine of QASs. Furthermore, ranking candidate passages is also a substantial task at the end of the QAS pipeline that can also affect QAS performance in the IR phase. It aims to re-rank the retrieved passages such that the most relevant ones appear first. In fact, PR is not only the core component of QASs but also constitutes an intermediate phase in many other tasks like text summarization [18] and the entity oriented search [6]. Owing to the fact that full documents can be too long and might cover different subjects, retrieving short and most specific text units in the documents might be of much merit [4].

In an attempt to enhance the performance of existing QASs and ensure the relevance of the selected passages, we focus on these two tricky tasks namely, passage retrieval and ranking. Basically, the architecture of a QAS and the techniques proposed often depend on some factors, mainly question domain and language [12]. In our work, we intend to give a broad approach that depends neither on the question domain nor on the user's language.

In view of this, in this paper, we propose the use of n-grams to retrieve passages and SVM to re-rank the retrieved passages for an open domain QAS. Our approach includes a PR engine relying on a new similarity measure, based on the dependency degree of n-gram words of the question in the passage. Thereafter, the retrieved passages are re-ranked by means of a Ranking SVM model that incorporates a set of metrics which constitute the features. In addition to our proposed n-gram measure, the features include other lexical, syntactic and semantic features which have already shown promise in the Semantic Textual Similarity task (STS) [7] which requires participating systems to determine the degree of similarity between pairs of text sentences and we have adapted these features to the context of QA. Our proposed approach aspires to automatically return the most relevant passage that is expected to contain the best answer to the given user's question.

The rest of this paper is worded as follows. In Section (2), we summarize the major existing studies that are related to both PR and ranking. Then, we introduce in Section (3) our approach and we detail its different steps. In Section (4), we move on to the description of our experimental study carried out to validate our proposed approach by means of the CLEF dataset and we compare our results with those of similar solutions performing the same task. Finally, in Section (5) concluding remarks and some perspectives are outlined.

2 Related Work

2.1 Approaches to Retrieving Passages

PR is an intermediate phase between document retrieval and answer extraction and it aims to reduce the search space for finding an answer by reducing the huge collection of documents into a fixed number of short paragraphs named passages. It is often viewed as a core component of a typical QAS. Hence, numerous approaches have been examined in the past for the purpose of PR in order to improve the performance of the QA task, where some of them are based on context. The most common context used for passages is their containing document such as in [15] where neighbor passages in the document are considered while distances between the question terms and the passage are ignored. Hyperlinks and the structure of an XML document [15] were also employed for passage contextualization in order to take into account word proximity in unstructured text such as in [19]. A further competitive contextualization approach was recently proposed by [9] using proximity scoring for focused retrieval. More concretely, a proximity value is attributed to each position in the text regarding its distance from the query terms. The proximity value of a position to a term is calculated based on the closest occurrence of the word to the position, while the proximity value of the query is the sum of the position proximity scores to all query terms applying fuzzy logic rules.

An early study carried out by [29] turned up that most proposed PR algorithms based on lexical matching between the query and the passages process each question term as an independent symbol and do not consider the order of words neither do their dependency relations in sentences. Moreover, several works rely on syntactic matching such as [11] mapping syntactic dependencies, also known as binary relations, instead of simple keywords to match the given question with the passages.

Thus, two main approaches were appeared, namely strict matching and fuzzy relation matching. The former one looks for an exact matching between the dependency relationships of the question and those of a passage at hand such as in [17], while the second one searches for a non exact matching between the dependency relationships of the question and those of a passage such as in [11].

Nonetheless, such models often require a syntactic parser which needs adaptation and its performance substantially affects that of the entire QAS. Furthermore, other works were based on semantic matching, seeking the meaning of terms, such as in [28] where various entity recognizers and semantic relatedness were employed based on MeSH [3]ontology and UMLS [4] semantic network to verify the relevance of candidate passages while [20] used FrameNet [5] frames to detect the semantic class of each term in the passage and the query. Despite the fact that semantic matching based approaches allow to find relevant answers, they often need semantic resources such as FrameNet which is currently available only in English and do not cover neither all domains nor all terms. Besides, there have been further subsequent works combining semantic and syntactic techniques in the context of PR such as [25] in order to take advantage of both techniques.

Beyond the simple lexical matching, n-grams structures which refer to contiguous sequence of terms extracted from a given , were introduced to improve the lexical matching between the question and the passages. N-grams have been widely used in natural language processing and have proven successful and effective in modern language applications, since they consider the dependency between terms rather than independent symbols. Within this context, a probabilistic method for web-based natural language QA was carried out by [23], where the web documents are segmented into passages to be ranked by means of an n-gram score based on tf-idf. Additionally, [10] developed a PR engine for QA based on an n-gram similarity metric where the passages including more question n-grams and longer ones are favored when comparing the candidate passages extracted over a set of documents using a keyword technique. Also, [8] followed the same previous approach using a different n-gram model which considers the passage n-grams existing in the question and their proximity. In a nutshell, our method in retrieving passages is different from the above n-gram based ones as we focus more attention on common n-grams between the question and the passage and we proceed otherwise to extract the n-grams.

[3]Medical Subject Headings thesaurus is a controlled vocabulary produced by the U.S. the National Library of Medicine.

[4]Unified Medical Language System is a thesaurus of biomedical concepts designed and maintained by the US National Library of Medicine. It consists of knowledge sources and a set of software tools and it also provides facilities for NLP.

[5]http://framenet.icsi.berkeley.edu/

2.2 Approaches to Ranking Passages

Question answering involves the extraction of relevant answers from sizeable documents, which begins by ordering the candidate passages retrieved by the PR module such that the most relevant ones are ranked first. Thus, significant research efforts have been conducted to solve this problem in QA. In fact, some passage ranking methods were based on knowledge such as [5, 3], where knowledge about question and answers is represented in the form of rules and inferences are derived from a knowledge base. However, the major drawback of such approaches is the need of a big number of inferences and manual design of features. Also, mapping and matching predicates and arguments to model relations remain extremely complex. Patterns are also widely used in several works such as [24] which are a collection of predefined context evaluation rules derived and designed from the question and the candidate answers but they often depend on the query type and domain. Additionally, pattern-based approaches approaches are often very complex and require a sizable collection of training data.

Otherwise, the context of words was used as a simple and intuitive method for ranking passages. For instance, [30] proposed a machine learning-based QA framework which incorporates context-ranking models named (CRM) that re-rank the passages retrieved from the initial retrievers to find the appropriate answers. The introduced model uses a Support Vector Machine (SVM) to combine contextual information of named entities, syntactic patterns and semantic features for each word in the context window as well as the word position to predict whether a candidate passage at hand is relevant to the question type. However, in the proposed framework, the performance of the CRM is highly dependent on the question classification. This latter work is most related to ours insofar as we have relied on SVM to rank the retrieved passages given a set of powerful lexical, syntactic and semantic text similarity measures rather than a simple detection of the NE class, the form and the part part-of-speech tag of the word. It is noteworthy to mention that SVM has previously proven to statistically outperform other machine learning methods in answer selection task for QA like decision tree learning (C4.5) boosting with decision tree learning (C5.0) and the maximum entropy method [27].

3 Approach Overview

In this section, we describe our proposed approach that is able to return the most relevant passage to a given user's question from a large-scale document collection. The overall architecture of our passage retrieval and re-ranking approach named PreRankQA is illustrated in Figure 1. Our primary principle is to first reduce the search space for finding an answer relying on n-gram structures to retrieve a small number of passages that are most likely to contain a correct response to the user's query.

We set the number of returned passages at 10 because major cited existing PR engines chose values near 10. However, we believe that n-gram technique is not enough to guarantee significant relevance, since it just considers simple dependencies between terms. Thus, in an attempt to improve the performance of our PR engine, we suggest to re-rank the retrieved passages using a Ranking SVM model that incorporates lexical, syntactic and semantic similarity measures in order to return the top ranking passage, as the most relevant one to the given question.

Basically, PreRankQA approach is composed of three main modules, namely question processing, PR and passage re-ranking. In the remainder of this section, we detail its different modules, mainly the last two ones concerned with our contribution.

3.1 Question Processing

Question processing is a fundamental task in any QAS and aims to process the natural language question posed by the user and extract some immediate information such as the question class as well as the answer type which may be useful in the answer extraction module. Our approach starts by preprocessing the user's question and extract the useful terms in order to generate a formal query. This latter is obtained by applying text

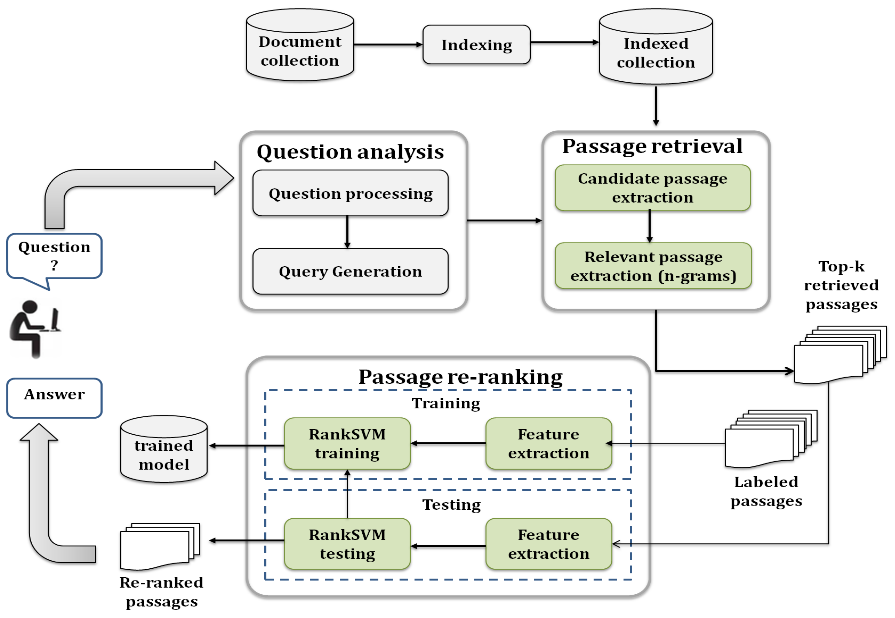

Fig. 1. Architecture of the PreRankQA approach

cleaning, question words elimination, tokenization, stop words removal and stemming. Thus, at the end of the question analysis module, we obtain a query formally defined as follows:

$$Q = \{t_1, t_2, ..., t_q\}, \quad (1)$$

where t denotes a separate term of the query Q and q represents the number of query terms.

3.2 N-gram based Passage Retrieval Module

Our PR module consists of two main phases: In the first one, we extract all passages containing the question terms, named candidate passages. Then, in the second phase, we filter these latter in order to keep the potentially relevant ones using n-gram structures. This Subsection is devoted to describing these phases.

3.2.1 Phase 1: Candidate Passage Retrieval:

Above all, we segment the document collection into paragraphs called passages. Obviously, a passage should be neither too short nor too long because, short excerpts might not contain the response, and long ones might include additional information that can skew the response or need more powerful extraction engine. To index the collection, for each passage, we save its id, number and text and other related information such as its document name. Thereafter, we extract their terms, apply the stemming and remove all the stopwords. A passage is formally defined as follows:

$$P = \{t_1, t_2, ..., t_p\}, \quad (2)$$

where t denotes a separate term of the passage P and p is the number of passage terms. Subsequently, we give a frequency to each passage term in order to calculate the maximum frequency and the term weights. The terms of indexed passages will be then stored in an inverted index.

In order to calculate the weight of the query terms, we rely on the formula 3 used in [10]:

$$w(t_i, q) = 1 - \frac{log(n(t_i))}{1 + logN}, \quad (3)$$

where n is the number of passages containing the term t and N is the total number of passages. This formula does not take into account the frequency of words but it only considers their discriminating power between passages. Thus, a term found in a single passage will have a maximum weight as the ratio value in the formula is low. Candidate passages are those that contain at least one of the query terms. To determine them, we just need to look for the query terms in the inverted index, where for each term the list of related passages is recorded and take the intersection of these passages. The candidate passages are defined as:

$$Pc = \{P_1, P_2, ..., P_n\}, \quad (4)$$

where P_i denotes a candidate passage, i its index and n the total number of candidate passages. Note that the weight of the candidate passages terms is computed in the same way as that of the query terms.

The candidate passages are filtered by calculating the similarity between each one and the question, using a similarity measure that only considers common words between the query and the passage, defined by equation 5 as follows:

$$s(p,q) = \frac{\sum_{t_i \in P \cap Q} w(t_i, q)}{\sum_{t_i \in Q} w(t_i, q)}. \quad (5)$$

Then, the candidate passages are ranked according to their similarity scores and their number (n) is reduced to (nb) which should be a happy medium between a big and a small number. Indeed, a big nb does not meet the goal mentioned above, but it has the advantage of not excluding passages that may contain the answer and are misclassified, while a small number can reduce the system complexity but with a strong chance of ignoring some passages. Recall that we intend to reduce the whole system complexity in terms of time and space and allow for a deeper analysis which was not possible before because of the massive collection size.

3.2.2 Phase 2: Relevant Passage Retrieval:

We highlight that the ultimate objective of our PR module is to identify the relevant passages from the document collection. To this end, the candidate passages returned by the previous phase should be filtered for a second time. Up until now, the extracted passages have only a few question words. However, this criterion is not good enough to judge the passage relevance. Thus, we should go beyond a simple verification of word occurrences. In this context, we intend to exploit other selection criterion such as the presence of word sequences, their length, their dependence. To this end, we use n-gram technique which can not only deal with a massive amount of data but also with its heterogeneity and it is further independent of language. In what follows, we detail the different steps to extract relevant passages.

N-gram Construction: Basically, we just focus on the common n-grams between the question and the passage. Thus, we start by identifying the common terms between a question and a passage, and then we derive their corresponding n-grams. We build the vector $\overrightarrow{Ct}$ of common terms between the question and the passage, by browsing through the terms of the question and check for each of them if it is also a term of the passage to add it in the vector. This latter is defined as follows:

$$\overrightarrow{Ct} \begin{pmatrix} t_1 & p_1Q & [p_{11}, .., p_{1m}] \\ t_2 & p_2Q & [p_{21}, .., p_{2m}] \\ .. & .. & \\ t_n & p_nQ & [p_{n1}, .., p_{nm}] \end{pmatrix},$$

where t_i is the ith term in common between the question and the passage with i=$\{1..n\}$. n is the number of question terms, p_iQ is the position of ith term in the question, p_{ij} is the jth position of the ith term in the passage and j=$\{1..m\}$. m is the passage terms number. Then, we construct the n-gram vectors of the question $\overrightarrow{NGQ}$ and the passage $\overrightarrow{NGP}$ by browsing the vector $\overrightarrow{Ct}$ and grouping the terms having successive positions in the question and the passage as follows: $\overrightarrow{NGQ} = \{ngQ_1, ngQ_2, ..., ngQ_q\}$ and $\overrightarrow{NGP} = \{ngP_1, ngP_2, ..., ngP_p\}$ where q is the number of query n-grams and p it that of the passage n-grams.

N-gram Weighting: The weight of each question n-gram is computed on the basis of its length and the sum of its term weights using formula 6:

$$w(ngQ) = l \times \sum_{t_i \in terms(ngQ)} w(t_i, q), \quad (6)$$

where l is number of terms contained in the question n-gram (ngQ). In fact, the multiplication of the weights sum by the n-gram length can foster adjacent words over the independent ones in the similarity calculation. We believe that grouped terms are more significant and less ambiguous than separate ones. Therefore, a term that belongs to an n-gram should have a greater weight than an independent one. As for the passage n-grams, they are weighted regarding their similarity degree with those of the question. We give a cumulative weight to the passage by browsing through the question n-grams and at each n-gram either its full weight or a lower one is added to the passage weight. In other words, if a question n-gram fully occurs in the passage, its whole weight will be added to the total weight, while if it is divided into smaller n-grams named subn-grams, a lower weight will be added to the cumulative weight which should be fixed according to the number of subn-grams in the passage. Hence, one among of the following cases might arise:

(a) The n-gram of the query is one of the passage n-grams.

(b) The n-gram of the query is made by combining a number n of the passage n-grams.

(c) The n-gram of the query is included in one of the passage n-grams.

Let w be the weight to add to the passage when we browse through the question n-grams ngQ. In the cases a and c, ngQ exists in the passage, so the additional weight w is set to: $w(ngP) = w(ngQ) = l \times \sum_{t_i \in terms(ngQ)} w(t_i, q)$ where l denotes the length of the n-gram ngQ and $w(t_i, q)$ is the weight of its term t_i. In the case b, ngQ is divided into subn-grams in the passage, let sng be the number of these subn-grams. In this case, the additional weight w is computed as follows:

$$w(ngP) = \frac{w(ngQ)}{sng} = \frac{l}{sng} \times \sum_{t_i \in terms(ngQ)} w(t_i, q). \tag{7}$$

NGsim Similarity Measure: Our n-gram based passage similarity measure named NGsim between a passage and a question is the ratio between the weight of the passage and that of the question. The passages are sorted according to this measure values to return those having the greatest values. The weight of a passage is is a sum of partial weights calculated at each step to be added to the total passage weight, while the query weight is calculated on the basis of its length and the sum of the weight of its terms. Given the weight of the question and that of each passage, we calculate $NGsim$ which is set to :

$$NGsim(p,q) = \frac{\sum_{i=1}^{q} \frac{l_i}{sng_i} \times \sum_{t \in terms(ngQ_i)} w(t,q)}{l(Q) \times \sum_{t_i \in (Q)} w(t_i, q)}, \tag{8}$$

where q is the number of the question n-grams, $l(Q)$ is the number of the question terms and $w(t_i, q)$ is the weight of a question term.

For example, let us consider the following terms of a question Q and those of a passage P:

Q(terms)= ***Presidency****,* ***European****,* ***Council****, vote,* ***Lisbon****,* ***Treaty****,* ***process****.*

P(terms)= ***Presidency****, regarding, message, benefits, project,* ***European****,* ***Council****, explaining, reasons, people,* ***Lisbon****,* ***Treaty****,* ***process****, Ireland, demonstrates, effort.*

The corresponding vector $\overrightarrow{Ct}$ of common terms between the given Q and P is set to:

$\overrightarrow{Ct(Q,P)}$= *Presidency, European, Council, Lisbon, Treaty, process*

From which we can derive $\overrightarrow{NGQ}$ and $\overrightarrow{NGP}$:

$\overrightarrow{NGQ(Q)}$ =[Presidency European Council][Lisbon Treaty process]
and
$\overrightarrow{NGP(P)}$ =[Presidency][European Council][Lisbon Treaty process].

In our example, $\overrightarrow{NGQ}$ is composed of two n-grams so we have two partial passage weights to compute. The first question n-gram is divided into two subn-grams in the passage so, sng equals 2 while the second one exactly equals a passage n-gram so, sng equals 1. So, given 1.632 and 1.536 the term weights of ngQ_1 and ngQ_2 respectively, $w_1(P) = \frac{l(ngQ_1)}{sng_1} \times \sum_{t \in terms(ngQ_1)} w(t,q) = (3/2) \times 1.632$ while $w_2(P) = (3/1) \times 1.536$. Thus, the total passage weight will be equal to the sum of $w_1(P)$ and $w_2(P)$. On the other hand, the question weight is set to: $w(Q) = l(Q) \times \sum_{t_i \in (Q)} w(t_i, q) = 8 \times 4.924$ where 4.924 is the result of the sum of query terms weight.

In summary, our approach for PR is different from the previously mentioned approaches based on n-grams. Indeed, we proceed differently to extract the n-grams, that is to say instead of extracting all n-grams for all n possible values of the question and the passage, as in [10] and [8], or all the n-grams of size n, as in [23], we extract only common n-grams between the question and the passages with different sizes. Hence, we do not need to include an additional step to select common n-grams from all the extracted n-grams. Additionally, for the weight of n-grams, we consider both the sum of the terms weight and their lengths like in [23], while [10] and [8] consider only the sum of the terms. Furthermore, we introduce a new similarity measure calculated from the weights of the question n-grams. We build a total passage weight by browsing the question n-grams, keeping the weight of an n-gram if it is found in the passage and reducing the weight of an n-gram if it is divided into smaller n-grams in the passage.

3.3 RankSVM based Passage Re-ranking Module

We emphasize that our PR module based on n-gram structures cannot guarantee high relevance of the retrieved passages since it can only ensure simple dependencies between terms. Thus, we propose to integrate other significant similarity measures combined by a Ranking SVM model referred to as RankSVM. This latter is a ranking version of SVM that was successfully applied in IR. It aims to solve the ranking problem in a supervised manner by transforming it into pairwise classification and then learn a prediction ranking function using the SVM principle. Basically, our passage re-ranking model consists of two phases: training and testing, where in both phases, the different metrics are computed for each passage and then entered into the RankSVM classifier to be re-ranked given their scores. Only the top ranked passage will be returned by the system as the most relevant answer to a given question. During the former phase, a set of annotated passages entered in the passage re-ranking model where each passage is labeled either Right (R) or Wrong (W) while in the testing phase, the passages are unlabeled since they are those extracted by our PR module. It is noteworthy that the features incorporated in our ranking model have already been proven successful in the Semantic Textual Similarity task [7](STS) at *SEM 2013 track which requires participating systems to determine the degree of similarity between pairs of text sentences. Among the proposed features described in [7], we will consider WordNet-based Conceptual Similarity, Named Entity Overlap, Edit distance, Syntactic Dependencies and instead of the N-gram based Similarity applied in this task that was not working very well, we resort to that proposed by ourselves in this work. We have adapted these features to the context of QA, where the sentence pairs become pairs of passage-question. Notice that we have picked out lexical, syntactic and semantic features to ensure answer relevance since, at this stage, text similarity measures based on term frequencies are not sufficient for retrieving relevant passages.

4 Experimental Study

4.1 Datasets and Measurement Metrics

Fundamentally, to evaluate a QAS, we require two main resources, namely a document collection and a question pool. For the evaluation of our PR engine, we used the dataset provided in the ResPubliQA 2009 exercise [22] of CLEF. The aim of this exercise is to retrieve paragraphs from the test collection to answer a question picked out from a set of 500 different questions falling into five types: factual, definition, reason, purpose and procedure. The document collection includes JRC-Acquis[6], where for each language, roughly 10700 documents are used containing over 1 million passages. In our experiments, we used that of English, Spanish and French and all the questions. On the other hand, in order to evaluate

[6]https://ec.europa.eu/jrc/en/language-technologies/jrc-acquis

the applicability of the whole approach, we used the resources proposed in the ResPubliQA2010 exercise [21] of CLEF. The objective of this exercise is to return either paragraphs or exact answers to a pool of 200 more complex questions from two test collections. The document collection includes a subset of JRC-Acquis as well as a small portion of the Europarl[7] collection made up of approximately 50 parallel documents. Notice that this time, we used the English collection because we employed the english versions of major tools deemed to achieve higher performance such as the english version of WordNet Lexical Database, but we can absolutely evaluate our approach in other languages by integrating multilingual tools. We emphasize that the given datasets based on passages ought to be the most suitable ones to validate our approach. We developed a system named PreRank (Passage REtrieval and RANKing) in Java using Eclipse environment and we have resorted to the open source system JIRS[8] (JAVA Information Retrieval System) described in [16] for indexing and search and SVM^{light} [9] for passage ranking and adapted them to our requirements. It is noteworthy that the training feeds annotated passages obtained from the judgment files of ResPubliQA2009 English collection.

The evaluation of the PR engine is based on the following measures which were intensively used in the QA task:

- The accuracy: is the percentage of correctly answered questions.
- The number of questions having correct passages ranked first.
- The Mean Reciprocal Rank (MRR): is the multiplicative inverse of the rank position of the first correct answer.

In order to evaluate the performance of the overall approach, we used the following measures proposed by CLEF:

[7]http://www.europarl.europa.eu/
[8]http://sourceforge.net/projects/jirs/
[9]http://svmlight.joachims.org/

- The c@1 measure: was proposed by CLEF as the main evaluation metric for passage and answer selection tasks, defined as: $c@1 = \frac{1}{n}(n_R + n_U\frac{n_R}{n})$ where n_R is the number of questions correctly answered, n_U denotes the number of questions unanswered and n is the total number of questions.
- The number of: unanswered questions (#NoA), questions correctly answered ($\#R$), questions wrongly answered (#W), unanswered questions where a right candidate answer is discarded (#NoA R) and questions unanswered with wrong candidate answer (#NoA W).

Recall that we have set a threshold value for the final score to be 0.15 as it is the most common used value for ranking. So, we answer the question only if the highest score value exceeds 0.15. Otherwise, we return no answer to the user's question, as we think that returning no response is better that delivering a wrong one.

4.2 Experiment Results

We compare our system results to those obtained by NLEL System [10] which includes a PR model based on n-grams and it was ranked first in the CLEF 2009 QA track for French and Spanish and second for English language. From Table 1, we see that PreRank has outperformed NLEL for the 3 languages on all criteria. A good number of questions was answered with a difference equals 17 questions more than NLEL for both French and Spanish languages and 24 for English. We obtained more answers in the first position with a difference between 18 and 25 questions. Also, the more the position increases, the less the number of question correctly answered is. Additionally, we obtained a greater MRR value for all languages because the number of correct answers is higher in the first positions and lower in the last ones. We can so admit that our NGsim measure is more efficient than that of NLEL.

Considering the whole approach, in Table 2, we compare the results yielded by our system run with those reported by similar systems performing the same task described in [21].

Table 1. Comparison between the PR module of PreRank and NLEL

Language	English		French		Spanish	
System	PreRank	NLEL	PreRank	NLEL	PreRank	NLEL
Number of questions having correct passages in the 10 first positions	298	274	282	265	284	267
Accuracy	0.851	0.782	0.805	0.742	0.811	0.762
Number of questions whose correct passage is in first position	186	161	171	153	174	154
MRR	0.453	0.377	0.411	0.367	0.413	0.373

Table 2. Comparisons of different systems on CLEF data

System	**PreRank**	**uiir**	**bpac**	**dict**	**elix**	**nlel**
Accuracy	**0.76**	0.72	0.68	0.67	0.65	0.64
c@1	**0.85**	0.73	0.68	0.68	0.65	0.65
#R	**152**	143	136	117	130	128
#W	**23**	54	64	52	70	68
#NoA	**25**	3	0	31	0	4
#NoA R	**0**	0	0	17	0	2
#NoA W	**0**	3	0	14	0	2

Table 2 shows that our system gives better results than all other ones in terms of both accuracy and $c@1$ with an accuracy value equals 0.76 and a $c@1$ score equals 0.85 which are significant results. Furthermore, the fact that our $c@1$ value is greater than the accuracy score proves that the use of our NoA criterion is justified. We have also remarked that most unanswered and incorrectly answered questions where opinion ones. Besides, we can reason out that the combination of various features has allowed to obtain relevant passages with high accuracy and c@1 values.

5 Conclusion and Future Work

In this paper, we have presented a language independent approach for retrieving and re-ranking passages for open domain QA using two powerful and effective models namely n-grams and SVM. We have evaluated the proposed approach by the development of the PreRank system and we have reported our findings. Even though our system have shown significant results over systems performing similar tasks, we emphasize that through our extensive experiments we have deduced that main preprocessing tasks such as tokenization, part of speech tagging and stemming take most of the response time. Thus, we expect in the future to improve the entire performance significantly by using parallel processing to speed up the preprocessing phase. Moreover, we have remarked that Wordnet similarity, Edit distance and N-gram measures were performing well, while Named Entity Overlap and Syntactic Dependencies measures were worst performing and time consuming. Therefore, we intend to improve these latter measures in order to reduce the system complexity and further enhance the re-ranking module performance. Finally, we look forward to extending our experiments on larger datasets to decide on the threshold value for the ranking final score.

References

1. **Abacha, A. B. & Zweigenbaum, P. (2015).** MEANS: A medical question-answering system

combining NLP techniques and semantic web technologies. *IPM*, Vol. 51, No. 5, pp. 570–594.

2. **Allam, A. M. N. & Haggag, M. H. (2012).** The question answering systems: A survey. *IJRRIS*, Vol. 2, No. 3.
3. **Araki, J. & Callan, J. (2014).** An annotation similarity model in passage ranking for historical fact validation. *Proc. of the 37th international ACM SIGIR conference on Research and Development in IR*, ACM, pp. 1111–1114.
4. **Arvola, P., Geva, S., Kamps, J., Schenkel, R., Trotman, A., & Vainio, J. (2010).** Overview of the inex 2010 ad hoc track. *International Workshop of the Initiative for the Evaluation of XML Retrieval*, Springer, pp. 1–32.
5. **Bilotti, M. W., Elsas, J., Carbonell, J., & Nyberg, E. (2010).** Rank learning for factoid question answering with linguistic and semantic constraints. *Proc. of the 19th ACM international conference on Information and KM*, ACM, pp. 459–468.
6. **Blanco, R. & Zaragoza, H. (2010).** Finding support sentences for entities. *Proc. of the 33rd international ACM SIGIR conference on Research and development in IR*, ACM, pp. 339–346.
7. **Buscaldi, D., Le Roux, J., Flores, J. J. G., & Popescu, A. (2013).** Lipn-core: Semantic text similarity using n-grams, wordnet, syntactic analysis, esa and information retrieval based features. *Proc. of the 2nd joint conference on LCS*, pp. 63.
8. **Buscaldi, D., Rosso, P., Gómez-Soriano, J. M., & Sanchis, E. (2010).** Answering questions with an n-gram based passage retrieval engine. *JIIS*, Vol. 34, No. 2, pp. 113–134.
9. **Carmel, D., Shtok, A., & Kurland, O. (2013).** Position-based contextualization for passage retrieval. *Proc. of the 22nd ACM international conference on Information & Knowledge Management*, ACM, pp. 1241–1244.
10. **Correa, S., Buscaldi, D., & Rosso, P. (2010).** NLEL-MAAT at RespubliQA. In *Multilingual Information Access Evaluation I. Text Retrieval Experiments*. Springer, pp. 223–228.
11. **Cui, H., Sun, R., Li, K., Kan, M.-Y., & Chua, T.-S. (2005).** Question answering passage retrieval using dependency relations. *Proc . of the 28th annual international ACM SIGIR conference*, ACM, pp. 400–407.
12. **Derici, C., Çelik, K., Kutbay, E., Aydın, Y., Güngör, T., Özgür, A., & Kartal, G. (2015).** Question analysis for a closed domain question answering system. In *Computational Linguistics and Intelligent Text Processing (CICLing2015)*. Springer, pp. 468–482.
13. **Etzioni, O. (2011).** Search needs a shake-up. *Nature*, Vol. 476, No. 7358, pp. 25–26.
14. **Faiz, R. (2006).** Identifying relevant sentences in news articles for event information extraction. *IJCPOL*, Vol. 19, No. 01, pp. 1–19.
15. **Fernández, R. T., Losada, D. E., & Azzopardi, L. A. (2011).** Extending the language modeling framework for sentence retrieval to include local context. *Information Retrieval*, Vol. 14, No. 4, pp. 355–389.
16. **Gómez, J. M., Buscaldi, D., Rosso, P., & Sanchis, E. (2007).** JIRS language-independent passage retrieval system: A comparative study. *Proc. of the 5th international conference on NLP (ICON-2007)*, pp. 4–6.
17. **Katz, B. & Lin, J. (2003).** Selectively using relations to improve precision in question answering. *Proc. of the workshop on Natural Language Processing for Question Answering (EACL 2003)*, pp. 43–50.
18. **Mihalcea, R. (2004).** Graph-based ranking algorithms for sentence extraction, applied to text summarization. *Proc. of the ACL 2004 on Interactive poster and demonstration sessions*, ACL, pp. 20.
19. **Norozi, M. A., Arvola, P., & de Vries, A. P. (2012).** Contextualization using hyperlinks and internal hierarchical structure of wikipedia documents. *Proc. of the 21st ACM international conference on Information and knowledge management*, ACM, pp. 734–743.
20. **Ofoghi, B. & Yearwood, J. (2009).** Can shallow semantic class information help answer passage retrieval? In *AI 2009: Advances in Artificial Intelligence*. Springer, pp. 587–596.
21. **Peñas, A., Forner, P., Rodrigo, Á., Sutcliffe, R. F. E., Forascu, C., & Mota, C. (2010).** Overview of respubliqa 2010: Question answering evaluation over european legislation. *CLEF 2010 LABs and Workshops, Notebook Papers*.
22. **Peñas, A., Forner, P., Sutcliffe, R., Rodrigo, Á., Forăscu, C., et al. (2010).** Overview of respubliqa 2009: Question answering evaluation over european legislation. In *Multilingual Information Access Evaluation I. Text Retrieval Experiments*. Springer, pp. 174–196.

23. Radev, D., Fan, W., Qi, H., Wu, H., & Grewal, A. (2005). Probabilistic question answering on the web. *JASIST*, Vol. 56, No. 6, pp. 571–583.

24. Severyn, A., Nicosia, M., & Moschitti, A. (2013). Building structures from classifiers for passage reranking. *Proc. of the 22nd ACM international conference on CIKM*, ACM, pp. 969–978.

25. Shen, D. & Lapata, M. (2007). Using semantic roles to improve question answering. *Proc. of EMNLP/CoNLL*, pp. 12–21.

26. Sun, H., Ma, H., Yih, W.-t., Tsai, C.-T., Liu, J., & Chang, M.-W. (2015). Open domain question answering via semantic enrichment. *Proc. of the 24th International Conference on WWW*, pp. 1045–1055.

27. Suzuki, J., Sasaki, Y., & Maeda, E. (2002). SVM answer selection for open-domain question answering. *Proc. of the 19th international conference on Computational linguistics-Volume 1*, ACL, pp. 1–7.

28. Tari, L., Tu, P. H., Lumpkin, B., Leaman, R., Gonzalez, G., & Baral, C. (2007). Passage relevancy through semantic relatedness. *TREC*.

29. Tellex, S., Katz, B., Lin, J., Fernandes, A., & Marton, G. (2003). Quantitative evaluation of passage retrieval algorithms for question answering. *Proc. of the 26th annual international ACM SIGIR conference*, ACM, pp. 41–47.

30. Yen, S.-J., Wu, Y.-C., Yang, J.-C., Lee, Y.-S., Lee, C.-J., & Liu, J.-J. (2013). A support vector machine-based context-ranking model for question answering. *JIS*, Vol. 224, pp. 77–87.

Nouha Othman received the B.Sc degree in Computer Science from Institut Supérieur de Gestion (ISG) of the University of Tunis, Tunisia, and two master's degrees in Computer Science from ISG and Polytech Nantes of the University of Nantes, France. Currently, she is a Ph.D. candidate in Computer Science at ISG Tunis. Her research interests include information retrieval, natural language processing and machine learning.

Rim Faiz obtained her Ph.D. in Computer Science from the University of Paris-Dauphine, LAMSADE Lab., in France. She is currently a Professor in Computer Science at the Institute of High Business Study (IHEC), LARODEC Lab., University of Carthage, in Tunisia. Her research interests include Information Retrieval, Big Data, Text Mining, Machine Learning, Natural Language Processing, and Semantic Web. She has published several papers and has served as PC member and reviewer for several international conferences and journals. Dr. Faiz is also responsible of the Master ”E-Commerce & Technological Innovation” and the Master ”Business Intelligence” at IHEC, University of Carthage.

Article received on 09/01/2016; accepted on 05/03/2016.
Corresponding author is Nouha Othman.