A Relevant Passage Retrieval and Re-ranking Approach for Open-Domain Question Answering

Nouha Othman*, Rim Faiz**

*Université de Tunis, Institut Supérieur de Gestion de Tunis, LARODEC, Tunisia
othmannouha@gmail.com
**Université de Carthage IHEC Carthage, LARODEC, Tunisia
rim.faiz@ihec.rnu.tn

Abstract. Question answering systems (QAS)s aim to directly return precise answers to natural language questions. Retrieving and re-ranking passages are viewed as the most challenging tasks in a typical QAS and still require non-trivial effort. In this paper, we propose a novel approach for retrieving and re-ranking passages using n-grams and SVM. Our n-gram based passage retrieval engine relies on a new measure of similarity between a passage and a question. The retrieved passages are further re-ranked using a Ranking SVM model combining different text similarity measures in order to return the most relevant passage to a given question. Our experiments and results have shown promise and demonstrated that our approach is competitive.

1 Introduction

Over the past few years, with the ongoing development in information technology, the amount of data has been increasing massively every day. Thereby, recent years have witnessed a burgeoning interest in question answering (QA) which is one of the major research area under Information Retrieval (IR) with applications mainly from Information Extraction (IE) and Natural Language Processing (NLP) (Faiz, 2006), where the main aim is to directly deliver a precise and concise answer to a question posted by the user in natural language from a sizable collection of documents or database (Voorhees, 2001). Indeed, QA domain falls into two categories: closed-domain QA which deals with questions under a specific domain such as biology and medicine, and open-domain QA which deals with general questions in various domains without any limitation. In our work, we focus on the second category as the techniques used are not tailored toward a specific domain. Basically, a typical QAS can be broadly viewed as a pipeline which consists of four main modules (Tellex et al., 2003): question analysis, document retrieval, passage retrieval and answer extraction, where each module has to deal with specific challenges. In particular, passage retrieval (PR) is always considered as the key task of a typical QAS since it allows to reduce the search space from a large collection of documents to a fixed number of passages. Obviously, QASs cannot find the right response to a given question, unless the answer exists in one of the retrieved passages. Therefore, it was widely studied over the past few years. Re-ranking passages is also a crucial task at the end of the QAS pipeline which aspires to order the retrieved passages such that the most relevant

ones appear first. With the purpose of enhancing the performance of existing QASs, we focus on these two tasks in an attempt to increase the number of returned correct answers and ensure their relevance.

In this paper, we propose a new approach for retrieving and re-ranking passages for an open domain QAS. Our passage retrieval module is based on an n-gram model where we suggest a novel measure of similarity between a passage and a question based on the degree of closeness or dispersion of n-gram words of the question in the passage. The retrieved passages are further re-ranked using a Ranking SVM model (Joachims, 2002) combining different text similarity measures that constitute the features. These latter include our new n-gram based measure as well as other lexical and semantic features which have already been proven successful in the Semantic Textual Similarity task (STS) at *SEM 2013 (Buscaldi et al., 2013). We intent to return the most relevant passage, the top ranking one, as a relevant answer to a given question.

The remainder of this paper is organized as follows: In Section (2), we give an overview of related work on PR and ranking tasks. Then, we present in Section (3) our proposed approach for retrieving and re-ranking passages for QA. Section (4) is devoted to evaluating our method in order to prove its ability. In Section (5) we conclude our paper and outline some perspectives.

2 Related Work on Passage Retrieval and Ranking Tasks

In this Section, we review the most important approaches in the literature related to PR categorized by the applied matching type between the candidate passages and the question, and the main proposed works on ranking for answer selection categorized according to the applied model.

2.1 Matching Types for Passage Retrieval

Indeed, PR is usually considered as the kernel of a typical QAS. Thus, numerous approaches have been developed for the purpose of PR to enhance the performance of QASs. Tellex et al. (2003) performed a quantitative evaluation of different PR algorithms for QA based on lexical matching where he deduced that most proposed algorithms process each term of the question as an independent symbol and do not consider the order of terms and their dependency relations. In order to address this shortcoming there have been several attempts to consider term dependencies. Furthermore, several works relied on syntactic matching like (Cui et al., 2005) deriving syntactic relations to match questions with passages. Nonetheless, the major limitation of syntactic matching is the need of a syntactic parser which is not always available for any languages. Added to that, it requires adaptation and the performance of the QAS significantly depends on the performance of the analyzer. Other works were based on semantic matching such as (Ofoghi and Yearwood, 2009). Although they allow to detect genuine answers, they require semantic resources which are usually not available in all languages and do not cover neither all domains nor all terms. Also, there have been further several works combining both semantic and syntactic techniques in the context of PR. Otherwise, some works resorted to a different model for retrieving passages going beyond the above simple lexical matching. These latter relied on n-grams which refers to sequences of consecutive words extracted from a given text, as a powerful and fast tool that does not deal with terms as independent symbols but it takes into account the simple dependency between them. In this

context, Radev et al. (2005) proposed a probabilistic method for web-based natural language QA, where the web documents are segmented into passages which will be ranked using an n-gram score based on tf-idf weighting. In addition, Correa et al. (2010) developed a PR system for QA based on an n-gram similarity measure favoring the passages containing more query n-grams and longer ones to compare the candidate passages extracted by key-words technique. Also, Buscaldi et al. (2010) followed the same previous approach but the only difference is about the n-gram model applied which considers the passage n-grams existing in the question and their proximity. Our method for retrieving passages is different from the above n-gram based ones, as we put more focus on common n-grams between the question and the passage and we proceed differently to extract the n-grams. Indeed, 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 fully found in the passage and reducing this weight if it is divided into smaller n-grams in the passage.

2.2 Ranking Approaches for Answer Selection

One of the key steps at the end of the QAS pipeline is answer selection which starts by ranking the candidate passages provided by the PR module. Given a query that expresses a user's question, the ranking function orders the retrieved passages such that the most relevant ones to the given query appear first. Thus, several ranking models have been proposed to tackle this problem in QA. In fact, some works relied on models based on knowledge to rank passages such as (Bilotti et al., 2010; Araki and Callan, 2014) where knowledge about question and answers such as named entity (NE) features, is represented as an annotation graph and inferences are derived from the knowledge base. Nevertheless, the major limitation of most knowledge-based approaches is that they rely on a huge number of inferences and manual design of features. Besides, mapping and matching predicates and arguments to model relations remains a very complex task. Other works introduced approaches based on patterns also called templates which are no more than a set of predefined context evaluation rules designed and extracted from the question and the candidate passages and they are often applied depending on the question type. For instance, Severyn et al. (2013) applied learning to rank models to automatically learn complex patterns, for instance, learning the relational semantic structures which appear in questions as well as their passages. However, most pattern based approaches are very complex and require a lot of training data. Otherwise, some works used the context of words as a simple intuition for ranking passages such as (Toba et al., 2010; Yen et al., 2013). In the latter work, authors proposed a context-ranking model named (CRM) that re-ranks the retrieved passages using contextual information of proper names, syntactic patterns, semantic features for each word in the context window as well as the word position to predict whether a given candidate passage is relevant to the question type. However, the performance of the CRM is highly dependent on the question classification. It is worth noting that our passage re-ranking model is broadly inspired from this latter approach based on SVM which is a supervised learning model that was in most cases successfully applied for passage ranking such as in (Moschitti and Quarteroni, 2011). Nonetheless, we resort to Ranking SVM model instead of SVM, which incorporates a set of features different from those used in (Yen et al., 2013). Unlike the latter, we are constrained neither by NE words nor by a fixed-size window. We went beyond a simple detection of the NE class, the form and the part part-of-speech tag of the word to calculate lexical, semantic and n-gram similarity measures.

3 Proposed Approach

The basic idea of our approach is to first reduce the search space using n-grams to retrieve the top 10 passages that are most likely to answer the user's question. The number of returned passages is set at 10 as most of the state of the art PR systems take values close to 10. However, since n-gram structure only relies on a simple dependency between terms, it is not enough to guarantee high relevance. Therefore, we tend to re-rank the retrieved passages using a Ranking SVM model that combines additional lexical and semantic similarity measures in order to return the most relevant passage, the top ranking one, to answer the input question. In this Section, we present our approach for retrieving and re-ranking passages named PRR which is basically composed of three main modules: question analysis, PR and passage re-ranking. Figure 1 depicts the overall architecture of the proposed PRR approach. In what follows, we detail its different constituents.

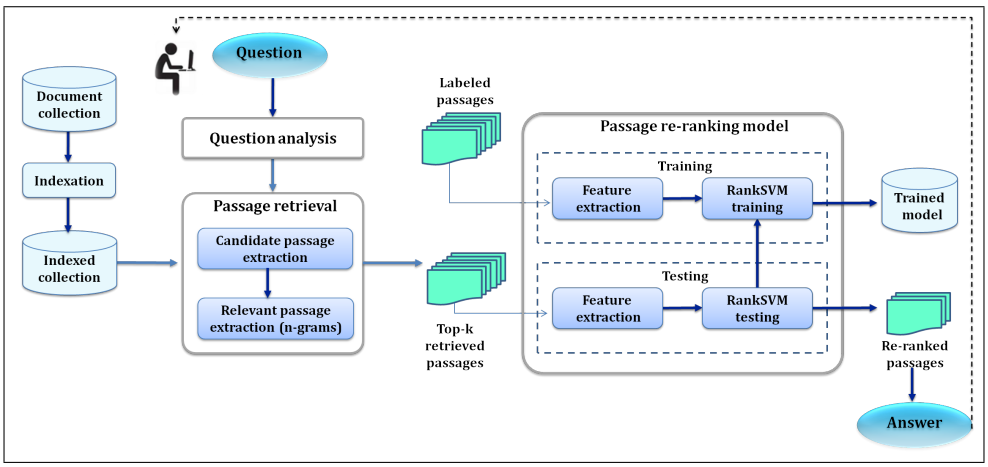

FIG. 1 – *Global architecture of our PRR approach*

3.1 Question Analysis Module

The intent of question analysis is to generate a formal query by preprocessing the question entered by the user and extracting the useful terms. This query is generated by applying text cleaning, question words elimination, tokenization, stop words removal and stemming. The query will be formally defined as: $Q = \{t_1, t_2, ..., t_q\}$ where t denotes a separate query term and q represents the number of terms in the query.

3.2 Passage Retrieval Module

Basically, our PR module is composed of two principle components: candidate passage extraction and relevant passage extraction. This Subsection is devoted to detailing them.

3.2.1 Candidate Passage Extraction

First of all, we index the collection by chopping the documents into paragraphs named passages. For each one, we save its related information such as its id, document name, number and text. Then, we extract the terms and eliminate the stopwords. Stemming is also applied to make easier the search and the word indexing. A passage is formally defined as: $P = \{t_1, t_2, ..., t_p\}$ where t represents a separate term of the passage P and p denotes the number of passage terms. Thereafter, we give a frequency to each passage term in the passage in order to compute the maximum frequency and the term weights. Once the passages are indexed, their terms will be stored in the inverted index. In order to calculate the weight of the query terms, we have resorted to the formula employed in (Correa et al., 2010) which does not consider the frequency of words but it only takes into account their discriminating power between passages. To identify the passages that contain at least one of the query terms, 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 named Pc are defined as follows: $Pc = \{P_1, P_2, ..., P_n\}$ where P_i is a candidate passage and n is the number of candidate passages. Note that the weight of a candidate passage terms is calculated by the same way as the query terms. In order to filter the candidate passages, we compute the similarity between each candidate passage and the question using the following similarity measure that only considers words in common between the query and the passage: $s(p,q) = \frac{\sum_{t_i \in P \cap Q} w(t_i,q)}{\sum_{t_i \in Q} w(t_i,q)}$. The candidate passages are then ranked according to their similarity values. Thus, the number of candidate passages (n) is reduced to (nb). The set of returned candidate passages will be set to: $Pc = \{P_1, P_2, ..., P_{nb}\}$ We ought to reduce the overall system complexity in terms of time and space and enable a deeper analysis which was not possible beforehand due to the huge collection size. Hence, the number nb should be fixed trying to find a happy medium between a big and a small number in order to reduce the system complexity but without excluding passages that may contain the answer and are misclassified. For example, we can set the number nb to 100 Next, we will apply another filter to the nb candidate passages using n-gram structures in order to extract the relevant ones.

3.2.2 Relevant Passage Extraction

Up until now, the candidate passages returned by the previous step have only a few number of words in common with the question. Nevertheless, this criterion is not good enough to judge the relevance of a passage. Therefore, it is necessary to get beyond a simple verification of word occurrences. Within this context, our proposed methodology aims to exploit other selection criteria such as the presence of word sequences, their length, their dependence, etc. For this purpose, we use n-gram technique which ensures simple dependency between terms, ambiguity reduction, language independence and it can not only deal with a huge amount of data but also with its heterogeneity. Our methodology for extracting relevant passages consists of the following parts:

N-gram Generation We first identify the common terms between a given question and passage and then we construct the vector $\overrightarrow{Tc}$ of common terms between the question and the passage. We just need to browse 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 by:

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

where t_i is the ith term in common between the question and the passage and i=$\{1..n\}$. n denotes the number of the question terms, p_iQ represents the position of the term i in the question, p_{ij} is the jth position of the ith term in the passage and j=$\{1..m\}$. m is the number of terms in the passage.Thereafter, we construct the n-gram vectors of the question vector $\overrightarrow{NGQ}$ and the passage $\overrightarrow{NGP}$ by browsing the vector $\overrightarrow{Tc}$ and grouping the terms having successive positions in the question and in the passage.

N-gram Weighting The weight of each n-gram of the question is calculated on the basis of its length and the sum of its term weights according to: $w(ngQ) = l \times \sum_{t_i \in terms(ngQ)} w(t_i, q)$ where l is number of terms contained in the question n-gram (ngQ). Indeed, the multiplication of the sum of the weights by the n-gram length can favor adjacent words rather than independent ones. Obviously, grouped words are more meaningful and less ambiguous than separate words. So, it is reasonable to give an independent word a greater weight when it belongs to an n-gram. As for the passage, the n-grams are weighted according to their similarity degree with the n-grams of the question. We assign a cumulative weight to the passage by browsing through the n-grams of the question and at each n-gram either is its whole weight or a lower one is added to the weight of the passage. More precisely, if an n-gram of the question is found in the passage, its full weight will be added to the cumulative weight, but if the n-gram is divided into smaller n-grams in the passage a lower weight will be added to the cumulative weight. This lower weight is fixed according to the number of the small n-grams. It is noteworthy that there are three possible cases:

- Case 1: The query n-gram is one of the passage n-grams: $ngQ_i = ngP_j, ngP_j \in NGP$
- Case 2: The query n-gram is made by combining a number n of the passage n-grams: $ngQ_i = \cup\, ngP, ngP \in NGP$
- Case 3: The query n-gram is included in one of the passage n-grams: $ngQ_i \in ngP_j$, $ngP_j \in NGP$

Let w be the weight to add to the passage when we browse through the question n-grams ngQ. In the cases 1 and 3, ngQ exists in the passage, so the additional weight w is calculated using the following formula: $w(ngP) = w(ngQ) = l \times \sum_{t_i \in terms(ngQ)} w(t_i, q)$ where l is the length of the n-gram ngQ and $w(t_i, q)$ is the weight of its term t_i. In the case 2, ngQ is divided into sub-n-grams in the passage, let sng be the number of these latter. In this case, the additional weight w is set to: $w(ngP) = \frac{w(ngQ)}{sng} = \frac{l}{sng} \times \sum_{t_i \in terms(ngQ)} w(t_i, q)$

Psim Similarity Measure Our passage similarity measure referred to as Psim between a passage and a question is no more than the ratio between the weight of the passage and that of the question. We order the passages according to the values of this measure in order to return those having the highest values. The weight of a passage is cumulative as mentioned above. It is a sum of partial weights calculated at each step to be added to the total passage weight. A

calculation step corresponds to the verification of the occurrence of an n-gram of the question into the passage and the underlying weight is given by the following formula 1:

$$w(P) = \sum_{i=1}^{q} \frac{l_i}{sng_i} \times \sum_{t \in terms(ngQ_i)} w(t,q) \quad (1)$$

where q denotes the number of the question n-grams. As for the weight of the question, it is calculated according to formula 2:

$$w(Q) = l(Q) \times \sum_{t_i \in (Q)} w(t_i, q) \quad (2)$$

where $l(Q)$ is the number of the question terms and $w(t_i, q)$ is the weight of a question term. $w(Q)$ is the same as the weight of an n-gram calculated using formula 3.2.2 when the n-gram ngQ is the question. $l = l(Q)$ is the number of the question terms and $sng = 1$ since all terms are grouped together and form a uni-gram. Once had the weight of the question and the weight of each passage, we calculate the similarity measure which is set to formula 3:

$$Psim(p,q) = \frac{w(P)}{w(Q)} = \frac{\sum_{i=1}^{q} \frac{l_i}{sng_i} \times \sum_{t \in terms(ngQ_i)} w(t,q)}{l(Q) \times \sum_{t_i \in (Q)} w(t_i, q)}. \quad (3)$$

Obviously, this similarity is maximum when all the terms question are grouped in the passage. For example, we consider the following terms of a question Q and those of a passage P:

Q(terms)= trade, ***ammonium****,* ***nitrate****,* ***fertilizers****, hampered,* ***European****,* ***Economic****,* ***Community****.*

P(terms)= ***ammonium****,* ***nitrate****, essential, ingredient, variety, products, some, intended, use,* ***fertilizers****, others, explosives, reason, divergencies, national, provisions, classification, content,* ***European****,* ***Economic****,* ***Community****, regulations, controlling, marketing .*

The corresponding vector $\overrightarrow{Tc}$ between the given question and passage is set to:

$\overrightarrow{Tc(Q,P)}$= *ammonium, nitrate, fertilizers, European, Economic, Community.*

From this latter we can deduce:

$\overrightarrow{NGQ(Q)}$ =[ammonium nitrate fertilizers][European Economic Community]

and $\overrightarrow{NGP(P)}$ =[ammonium nitrate][fertilizers][European Economic Community].

In this example, $\overrightarrow{NGQ}$ is composed of two n-grams so we have two partial passage weights to calculate. The first question n-gram is divided into two subn-grams in the passage so, sng equals 2 while the second one is exactly equal to a passage n-gram so, sng equals 1. Thus, given 1.766 and 1.524 the term weights of ngQ_1 and ngQ_2 respectively, $w_1(P) = \frac{l(ngQ_1)}{sng_1} \times \sum_{t \in terms(ngQ_1)} w(t,q) = (3/2) \times 1.766$ while $w_2(P) = \frac{l(ngQ_2)}{sng_2} \times \sum_{t \in terms(ngQ_2)} w(t,q) = (3/1) \times 1.524$. So, the total passage weight will be equal to the sum of $w_1(P)$ and $w_2(P)$. On the other hand, the weight of the question will be 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.

To sum up, our method is different from the previously cited n-gram based ones. First, we proceed differently to extract the n-grams insofar as instead of extracting all n-grams for all n possible values of the question and the passage, as in (Correa et al., 2010) and (Buscaldi et al., 2010), or all the n-grams of size n, as in (Radev et al., 2005), we extract only common n-grams between the question and the passages with different sizes. So, we do not need to include

an additional step to select common n-grams from all the extracted n-grams. Second, for the weight of n-grams, we take into account both the sum of the terms weight and their lengths like in (Radev et al., 2005) while Correa et al. (2010) and Buscaldi et al. (2010) consider only the sum of the terms weight. Third, our $Psim$ measure is computed from the weight of the question n-grams and depends on whether the n-gram is fully exists or it is divided in the passage.

3.3 Passage Re-ranking Module

Given a set of retrieved passages returned by our n-gram based PR engine, we resort to the Ranking SVM referred to as RankSVM model which combines different text similarity measures that constitute the features to re-rank them according to their similarity degrees with the question. RankSVM is a ranking version of the SVM model introduced to solve the ranking problem in a supervised manner. The underlining idea behind this popular machine learning method is to transform the ranking problem into pairwise classification and then learn a prediction ranking function using the SVM intuition. We point out that RankSVM has been successfully applied in the context of IR, in particular to document retrieval (Cao et al., 2006). It is noteworthy that our passage re-ranking model consists of two phases: training and testing. In both phases, the different similarity measures are calculated for each passage and then these latter are entered into the RankSVM classifier which will re-rank the passages given their feature values. Only the passage ranked first by our model will be returned by the system as the most relevant answer to a given user's question. During the first phase, a set of annotated passages entered in the passage re-ranking model to ranked where each passage is labeled either +1 (right) or -1 (wrong), while in the testing phase, the passages are not labeled as they are those extracted by our PR module. Although n-gram technique ensures only a simple dependency between terms, it seems not satisfactory enough to ensure the relevance of the extracted passages. Thus, we resort to other significant features, particularly the semantic ones. In fact, the features employed in our ranking model have already been proven successful in the Semantic Textual Similarity task (Buscaldi et al., 2013)(STS) at *SEM 2013 which requires participating systems to determine the degree of similarity between pairs of text sentences. Among the proposed features, we will only consider WordNet-based Conceptual Similarity, Named Entity Overlap, Edit distance and instead of the N-gram based Similarity applied in this task we resort to that proposed by ourselves in this work. Indeed, we have adapted these features to the context of QA. That is to say, the sentence pairs become pairs of passage-question. Note that the more we add features, the higher the program complexity is. Moreover, at this stage, text similarity measures based on term frequencies are not sufficient for retrieving relevant passages (Keikha et al., 2014). Whence, in our case we have mainly resorted to semantic features to ensure answer relevance.

4 Experimental Evaluation

4.1 Datasets and Tools

Basically, two main resources are required for the development and the evaluation of a QAS: a document collection and a question pool. For the evaluation of our PR module, we

have used the dataset provided in the ResPubliQA 2009 exercise (Peñas et al., 2010) of CLEF which seeks to retrieve paragraphs from the test collection to answer a given question picked out from a set of different questions. In our experiments, we will use the French collection as only few systems have been tested in this language due to its ambiguity. The experiments are carried out on 338 questions, 1388818 passages derived from the given collection, 100 passages used from those returned by the search model and 10 passages returned by the n-gram model. On the other hand, to assess the applicability of the overall approach, we have used the resources provided in the ResPubliQA2010 exercise (Peñas et al., 2010) which aims to return either paragraphs or exact answers as system output to a set of 200 more complex questions over two test collections. Note that this time, we will use all the English collection for training and testing. In fact, as we work on passages, the given datasets ought to be the most appropriate ones to evaluate our approach. We emphasize that we have tested the overall approach using the english corpora as we have resorted to the english versions of major used tools deemed to achieve higher performance such as the english version of WordNet Lexical Database and the NE recognizer for english. Obviously, we can also evaluate our approach in other languages merely by integrating multilingual tools. In order to validate our approach, we have developed a system named PexRank (passage extraction and ranking system) and we have used the open source system JIRS [1](JAVA Information Retrieval System) described in Gómez et al. (2007) for indexing and search processes and adapted it to our need. For passage ranking we have used the open source SVM^{light} [2]. Note that the training feeds labeled passages derived from the judgment files of ResPubliQA2009 English question/answer pairs.

4.2 Evaluation Metrics

To evaluate the performance of our PR engine, we are based on the following measures:

- The accuracy: which is defined as the percentage of correct answers compared to all asked questions (in our case for the first 10 positions).
- The number of questions having correct passage ranked first.
- The Mean Reciprocal Rank (MRR) which denotes the multiplicative inverse of the rank position of the first correct answer and was widely been used in QA for answer ranking.

To evaluate the whole approach, we are based on the following CLEF measures:

- The c@1 measure which was introduced by CLEF as the major evaluation measure for both passage and answer selection tasks. The formula of $c@1$ is set to: $c@1 = \frac{1}{n}(n_R + n_U \frac{n_R}{n})$ where n_R denotes the number of questions correctly answered, n_U is the number of questions unanswered and n is the total number of questions.
- Overall accuracy: the accuracy calculated over all assessed answers.
- We also used the number of questions unanswered (#NoA), the number of questions answered correctly ($\#R$), the number of questions answered wrongly (#W), the number of questions unanswered where a right candidate answer is discarded (#NoA R): In this case, the system chooses to leave the question unanswered (pessimistic behavior) and the number of questions unanswered with wrong candidate answer (#NoA W).

Notice that we have set a threshold value for the final score to be 0.15 as it has been chosen by many authors for the final ranking result. So, we answer the question only if the highest score

1. http://sourceforge.net/projects/jirs/
2. http://svmlight.joachims.org/

value exceeds 0.15. Otherwise, we do not return any answer to the given question. We believe that returning no answer to a posted question is better that delivering a wrong response.

4.3 Results and Discussion

We have compared the results obtained by our PR engine to those of NLEL system (Correa et al., 2010) which used an n-gram based PR model and it was ranked first in the CLEF 2009 QA track for the French language.

TAB. 1 – *Results of comparison between PexRank and NLEL*

	PexRank	NLEL
Number of questions having correct passages in the first 10 positions	**272**	260
Accuracy	**0.804**	0.670
Number of questions whose correct passage is in first position	**159**	142
MRR	**0.409**	0.365

Table 1 shows that our system has given better results than NLEL in all criteria. Indeed, PexRank has answered a significant number of questions, with a difference equal to 12 questions more than NLEL. We obtained more answers in the first position with a difference equal to 17 questions. Additionally, we deduced that the MRR value of PexRank is greater than that of NLEL because the number of answers obtained by our system is greater in the first positions and lower in the last ones. Now, we move on to evaluate the whole approach. The results obtained in our system run are presented in Table 2 where we compare our results to those reported by other systems participating in the ResPubliQA2010 exercise performing the same task of selecting the most relevant passage to answer a given question, described in (Peñas et al., 2010).

TAB. 2 – *Comparison between PexRank and similar systems*

System	Accuracy	c@1	#R	#W	#NoA	#NoA R	#NoA W
PexRank	**0.74**	**0.83**	**149**	**26**	**25**	**0**	**0**
uiir101PSenen	0.72	0.73	143	54	3	0	3
bpac102PSenen	0.68	0.68	136	64	0	0	0
dict102PSenen	0.67	0.68	117	52	31	17	14
bpac101PSenen	0.65	0.65	129	71	0	0	0
elix101PSenen	0.65	0.65	130	70	0	0	0
nlel101PSenen	0.64	0.65	128	68	4	2	2
uned102PSenen	0.65	0.65	129	71	0	0	0

The results reported in Table 2 show that PexRank outperforms all the other systems performing the same task in terms of both accuracy and $c@1$ measures with an accuracy score equal to 0.74 and a $c@1$ score equal to 0.83 which are significantly good results. Moreover, the fact that our $c@1$ value is greater than the accuracy score proves that the use of our no answer criterion is justified and has allowed obtaining a high $c@1$ measure. We have remarked that out of 99 complex questions, our system has successfully answered 48 questions. Most of the unanswered questions where opinion questions. Similarly, the incorrectly answered questions were mostly opinion and causes ones. Nonetheless, there is scope for further experiments on

larger datasets to decide on the threshold value for the ranking final score. Note that, We have chosen not to provide any candidate answer for unanswered questions, neither correct nor incorrect. Although the paragraph selection task is just a PR, the major difference from pure IRs is to add the option of leaving the question unanswered in the validation step. It is noteworthy that this task allows posting complex questions and evaluating them in a simple way.

5 Conclusion

Asking a question in natural language and having a precise answer becomes today a major asset to the user and it constitutes a considerable challenge in many application areas such as e-commerce, distance learning and mobile search. In this paper, we have tackled two crucial tasks in QA and proposed a new approach for retrieving and re-ranking passages taking advantage of n-grams and RankSVM models. Although our experimental results have shown promise, we believe that our approach could be improved by incorporating other features such as the syntactic one into the re-ranking model without significantly increasing the program complexity. In the future, we look forward to evaluating our approach on corpus written in other languages and to enabling our system to return a precise answer rather than a passage.

References

Araki, J. and J. Callan (2014). An annotation similarity model in passage ranking for historical fact validation. In *Proc. of the 37th international ACM SIGIR conf on Research and Development in IR*, pp. 1111–1114. ACM.

Bilotti, M. W., J. Elsas, J. Carbonell, and E. Nyberg (2010). Rank learning for factoid question answering with linguistic and semantic constraints. In *Proc. of the 19th ACM international conf on Information and KM*, pp. 459–468. ACM.

Buscaldi, D., J. Le Roux, J. J. G. Flores, and A. Popescu (2013). Lipn-core: Semantic text similarity using n-grams, wordnet, syntactic analysis, esa and information retrieval based features. In *Proc. of the 2nd joint conf on Lexical and Computational Semantics*, pp. 63.

Buscaldi, D., P. Rosso, J. M. Gómez-Soriano, and E. Sanchis (2010). Answering questions with an n-gram based passage retrieval engine. *JIIS 34*(2), 113–134.

Cao, Y., J. Xu, T.-Y. Liu, H. Li, Y. Huang, and H.-W. Hon (2006). Adapting ranking svm to document retrieval. In *Proc. of the 29th annual international ACM SIGIR conf on Research and Development in IR*, pp. 186–193. ACM.

Correa, S., D. Buscaldi, and P. Rosso (2010). Nlel-maat at respubliqa. In *Multilingual Information Access Evaluation I. Text Retrieval Experiments*, pp. 223–228. Springer.

Cui, H., R. Sun, K. Li, M.-Y. Kan, and T.-S. Chua (2005). Question answering passage retrieval using dependency relations. In *Proc. of the 28th annual international ACM SIGIR conf on Research and Development in IR*, pp. 400–407. ACM.

Faiz, R. (2006). Identifying relevant sentences in news articles for event information extraction. *IJCPOL 19*(01), 1–19.

Gómez, J. M., D. Buscaldi, P. Rosso, and E. Sanchis (2007). Jirs language-independent passage retrieval system: A comparative study. In *Proc. of the 5th international conf on NLP (ICON-2007)*, pp. 4–6.

Keikha, M., J. H. Park, W. B. Croft, and M. Sanderson (2014). Retrieving passages and finding answers. In *Proc. of the 2014 Australasian Document Computing Symposium*, pp. 81. ACM.

Moschitti, A. and S. Quarteroni (2011). Linguistic kernels for answer re-ranking in question answering systems. *IPM 47*(6), 825–842.

Ofoghi, B. and J. Yearwood (2009). Can shallow semantic class information help answer passage retrieval? In *AI 2009: Advances in Artificial Intelligence*, pp. 587–596. Springer.

Peñas, A., P. Forner, Á. Rodrigo, R. F. E. Sutcliffe, C. Forascu, and C. Mota (2010). Overview of respubliqa 2010: Question answering evaluation over european legislation. In *CLEF 2010 LABs and Workshops, Notebook Papers, 22-23 September 2010, Padua, Italy*.

Peñas, A., P. Forner, R. Sutcliffe, Á. Rodrigo, C. Forăscu, I. Alegria, D. Giampiccolo, N. Moreau, and P. Osenova (2010). Overview of respubliqa 2009: Question answering evaluation over european legislation. In *Multilingual Information Access Evaluation I. Text Retrieval Experiments*, pp. 174–196. Springer.

Radev, D., W. Fan, H. Qi, H. Wu, and A. Grewal (2005). Probabilistic question answering on the web. *JASIST 56*(6), 571–583.

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

Tellex, S., B. Katz, J. Lin, A. Fernandes, and G. Marton (2003). Quantitative evaluation of passage retrieval algorithms for question answering. In *Proc. of the 26th annual international ACM SIGIR conf on Research and Development in IR*, pp. 41–47. ACM.

Toba, H., S. Sari, M. Adriani, and R. Manurung (2010). Contextual approach for paragraph selection in question answering task. In *CLEF (Notebook Papers/LABs/Workshops)*.

Voorhees, E. M. (2001). The trec question answering track. *JNLE 7*(04), 361–378.

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

Résumé

Les systèmes de questions-réponses (SQR)s visent à retourner directement des réponses précises à des questions posées en langage naturel. L'extraction et le reclassement des passages sont considérés comme les tâches les plus difficiles dans un SQR typique et exigent encore un effort non trivial. Dans cet article, nous proposons une nouvelle approche pour L'extraction et le reclassement des passages en utilisant les n-grammes et SVM. Notre système d'extraction de passages basé sur la technique des n-grammes repose sur une nouvelle mesure de similarité entre un passage et une question. Les passages extraits sont ensuite réordonnés en utilisant un modèle basé sur RankSVM combinant différentes mesures de similarité afin de retourner le passage le plus pertinent pour une question donnée. Nos expériences et nos résultats étaient prometteurs et ont démontré que notre approche est concurrentielle.