Answer Selection in Question/Answering

Hugo Patinho Rodrigues

Instituto Superior Técnico, Technical University of Lisbon
Av. Rovisco Pais, 1 – 1049-001 Lisboa, Portugal
hugo.p.rodrigues@ist.utl.pt

Abstract

This thesis addresses the problem of answer selection in Question Answering (QA). Answer selection is one of the main tasks for QA and the best answer to be returned is selected in this step. For that, we propose AnSelMo, an ANSwering SELection MOdule based on the context where candidate answers appear. We tested AnSelMo in three distinct scenarios: ‘Who Wants to Be Millionaire?’, the famous contest of multiple-answer questions, Question Answering for Machine Reading Evaluation (QA4MRE), a Cross Language Evaluation Forum reading comprehension task, and Just.Ask, the L²F QA system. Results for WWBM surpass the state of the art, while for QA4MRE results are better than most of the 2011 competing systems’ results. We were also able to improve Just.Ask accuracy, by integrating AnSelMo on it.

Keywords: Question Answering, Answer Selection, Context, Word Proximity, Similarity Measures, Semantic Spaces

1 Introduction

Question Answering (QA) systems try to answer questions posed in natural language, in contrast to search engines, that return a set of related documents given some keywords. This is challenging because natural language is a powerful tool and questions may be formulated in different forms. Also, returning a single and concise answer is a much harder problem than returning a set of more or less related documents. This is done by QA systems in the answer selection step, after extracting possible candidate answers and rank them. The focus of this thesis is precisely this task where, given a set of candidate answers – which may or not be related with each other –, we have to choose one or more correct answers.

This problem is present in several QA systems, as in Just.Ask [1, 21, 10], the L²F QA system. Currently, Just.Ask already explores the problem of relating candidates [11], but it does not take into consideration the context in which the candidate appears. Considering the following question: What is the capital of Portugal? and the two candidate answers Lisboa and Porto, extracted from Lisboa, capital of Portugal and Porto was chosen, in Portugal, as the European Capital of Culture, respectively. Given the context, one could rule out the second candidate in favor of the first one due to the distance between the question keywords, capital and Portugal, and the candidate answers.

This problem can also be found in other scenarios in the Natural Language Processing (NLP) field. For example, QA4MRE, a task introduced in Cross Language Evaluation Forum (CLEF) in 2011 [15], aims at evaluating a system reading capability through multiple-answer questions. Briefly, given a text and some questions about it, competing systems have to choose the correct answer to those questions, among five candidates, showing in this way their level of “comprehension” of the text. The information needed to correctly choose between the different questions can be found in the given texts. The task also provides a Background Collection, which consists in a collection of documents that can help systems answer the questions.

Another similar scenario, already studied by the NLP community, is ‘Who Wants to Be Millionaire?’ (WWBM), the famous multiple-answer question contest. In this case, contestants face different questions, with four hypothesis of answer.

These two distinguish from the answer selection problem in QA for two reasons: (1) they always have a correct candidate answer; and (2) the candidate answers are not related, as opposite to the candidate answers extracted, for instance, by Just.Ask. Thus, these scenarios can be seen as a subproblem of the answer selection problem, as there are only up to five candidates, non-related, and where one and only one is the correct answer. However, the two are also distinct scenarios, due to their nature and complexity (QA4MRE questions tend to be trickier, as it has a Machine Reading (MR) task associated with it).

We follow two different approaches to the answer selection task. The first is based on two works from the beginning of the century, applied to WWBM, that we recover, modify and extend for this purpose. Both are directly related with the context of the candidate answer, and use different techniques to measure it. The second one is based on semantic spaces, and tries a different approach to the problem. The idea is to discover latent topics from documents, and associate the question and candidate answers to those topics, distinguishing them in the process.

The work of this thesis has as contributions (1) the development of a set of techniques that consider the context of candidate answers, from where they are extracted, in order to select them; and (2) an answer selection module, AnSelMo, based on those techniques.

This document is structured as follows: in Section 2 we present related word; Section 3 describes our system and its techniques, followed by the experiments performed, detailed in Section 4. Finally, in Section 5, we present the conclusions of this work and point to some future work.

2 Related Work

To our knowledge, the first computational attempt to solve WWBM is the one described by Clarke et al. [5], where the authors present an approach on the QA task (used at
TREC [4] and, later, on WWBM. Here, a query (a set of terms) is generated from the original question plus each one of the possible answers. The first task described by the authors is to find the so-called extents: passages from a document, containing at least once each term of the original query. These extents are classified and scored, considering the self-information contained in each extent. This approach presupposes a probability associated with each extent, which is given by the probability of finding each term of the query in the extent\(^1\). A higher score is given to extents with lower probability of occurrence. Higher values do not imply a greater likelihood of the answer being present in the extent or its proximity, but the authors found empirically that this relation holds. However, in the context of WWBM an answer is one or more terms, not an whole extent; thus, by following a similar ranking strategy, each term is scored, based in a redundancy component, representing the number of occurrences of each term in the different passages. The accuracy accomplished for a 108-question test was 70%.

In 2003, Lam et al. [8] made another approach to WWBM. The strategy is simple, yet with good results. The web is used as the only resource, and three different approaches are reported, with ranging accuracies from 50 to 75%. The first technique (the baseline) comes out directly from the problem formulation: the correct answer is the one with more results retrieved by querying a search engine with a query in the form “answer . questionModified”, where “questionModified” refers to the question filtered of stopwords. Also, for questions with negative constraints such as Which of these plays is not written by Shakespeare? A: Hamlet B: Othello C: Romeo and Juliet D: Cats, the chosen answer was the one with less results (an inversion due to the presence of not). Using Google the accuracy was set in about 60%.

The second strategy cares about the position of the answer in a given document, when compared with other words present in the question. Therefore, authors implemented a word-proximity algorithm, which relies on the assumption that answers appear close to the terms contained by the question, giving this way more weight to them. The reason behind this heuristic was the belief that – as in Clarke et al. [5] – answers appear in documents close to question words. It is intended to weigh the distances, of a maximum radius, so that documents with too many references to an answer but not to the corresponding question words are worth less. Different values for radius were tested and authors concluded that the optimal value for radius is not trivial to get. When combining different parameterizations, accuracies reach 75%.

Regarding QA4MRE, twelve systems participated in the 2011 track; however, only eight working notes are available [15]. For all submitted runs (43 for English, 11 for German and 9 for Romanian), nearly half reported a score below the baseline, which was of 20% in c@1, meaning random guesses to all questions. The winning system, Pakray et al. [14], achieved results of 48.3% accuracy (0.57 considering the c@1 measure).

Although very different approaches were followed by participating systems, several steps were common to many of them, and various resources were widely exploited. In fact, many systems performed pre-processing, namely: anaphora or co-reference resolution [3, 23]; stopword filtering [20] and Named Entity (NE) Recognition (NER) in the questions [7]. Regarding tools and resources, Lucene\(^2\) was used by many systems to index the texts (as described in Iftene et al. [7] and Martinez-Romo and Araujo [9]) and WordNet [12] was a constant presence – for instance Saias and Quaresma [20] report its use in synonym detection. Nevertheless, different strategies were implemented, from information retrieval- to logic-based approaches. For instance, Verberne [23] took advantage of the BM25 function to rank passages from the Background Collection according to their similarity to a given text fragment, expanding it in this way. In the work described by Iftene et al. [7], Lucene is also used to index the texts. The built index is then queried using the questions, creating this way another index, built with the retrieved passages/documents. Then, based on this new index, answers are used as queries in a new retrieval step. The relevance scores from each retrieval step are then used to compute a final score for each answer. Babych et al. [2], on the other hand, describe a system for German based in logical inferences. Here, given the candidate answer C, the input text T, and the Background Collection B, the system tries to infer if \((T \land B) \vdash C\). Text is parsed in a dependency graph, and hyponym and other relations are extracted from this graph. Other strategies try to relate terms. Saias and Quaresma [20] report the use of rules to measure the distance between the key elements of the question and the answer. The system described by Cao et al. [3] tries to simulate the strategy applied by people when learning a new language and answering reading tests. According to these authors, people will first locate named entities in the passages related with the questions. Thus, their system performs NER to find related passages and, afterwards, compares the NEs between the question and the passages. Terms are also related by using WordNet relations, such as synonym and hypernym. Each type of relation has a weight associated, which contributes to the final score.

In what concerns the winning system, Pakray et al. [14], it combines two different strategies: an Answer Validation (AV) approach and a QA approach. The best results were accomplished by using the system as an hybrid between the two. The AV module is based on textual entailments: for each answer of a given question, an hypothesis \(H\) is generated, according to a set of patterns. These are then used to retrieve passages from the texts, which are indexed with Lucene. The topmost sentence, \(T_s\), is paired with the corresponding hypothesis, resulting in the pair \((T_s, H)\). These pairs are then processed by a pipeline of different strategies to check if they are textual entailments. Among these strategies are the comparison of NEs, the number of co-occurring unigrams, bigrams and skip-bigrams between \(T_s\) and \(H\), and the matching of question and answer types. In what

---

\(^1\) An assumption of independence of each term appearing in the snippet is made.

\(^2\) http://lucene.apache.org/
concerns the QA module, it starts by doing a similar task. Following some rules, each question is transformed into a pattern, where the wh-word is substituted by one of the candidate answers. From these patterns are also extracted stopwords, creating a keyword list. Then, each pattern will be compared against each sentence from the documents. If they do not match, the same is done between the respective keywords list and the sentences. Whichever matches, a score is assigned.

Exploring other techniques that could be applicable, we found Latent Semantic Analysis (LSA), a mathematical technique first applied to Information Retrieval (IR) [6] that has been also applied to different areas, such as summarization [18, 13]. LSA can help to identify latent topics from documents, even without knowing them apriori.

LSA is a vector-space approach, based on Singular Value Decomposition (SVD) to reduce dimension from the original matrix $A$. This matrix is usually a term by document matrix, but sentences instead of entire documents may also be used. The matrix has dimensions $T \times N$, with $T$ being the total number of terms and $N$ the total number of documents. The matrix is filled with the scores of each term in the given document. These scores usually have a local and global component (for instance, $tf.idf$ score may be used), but only local weights such as term frequencies or other features may be used. From SVD application to matrix $A$ results a decomposition into three matrices:

$$A = U \Sigma V^T,$$

with $U$ being a $T \times m$ matrix, $\Sigma$ a $m \times m$ diagonal matrix, whose elements are non-negative singular values sorted in descending order and $V$ a $m \times N$ matrix. Figure 1 depicts the transformation.

![Figure 1: Application of SVD to matrix $A$.](image)

We can see that one can now define terms by topics (or concepts), and also topics by document. This allows another view on the documents, For example, in summarization one can extract only a subset of topics from a document and then select passages from that topic. This decomposition allows different analysis. For example, one can compare two documents by evaluating their similarity. This is done by comparing their vectors (which are normalized due to SVD), applying, for instance, a cosine function between them. There are similar approaches, using the same paradigm (semantic spaces and dimension reduction), but that rely in probabilities instead of frequencies. These are called Probabilistic Topic Models, and two examples are pLSA (LSA with a probabilistic component) and LDA (Latent Dirichlet Allocation), which differs by adding a Dirichlet prior on the topic distribution [22].

3 AnSelMo

Following some of the work presented in last section, we built a system, as much as language independent as possible, that aims at answering multiple-answer questions. This section presents such system, AnSelMo (ANSwering SELection MOdule), starting with its architecture, followed by a dedicated section to each one of its modules: Pre-Processing, Counting, Lexical Approaches, Latent Semantic Analysis and Scoring.

3.1 Architecture

Figure 2 depicts the system’s architecture. The core is the column of strategies that can be used to perform the answer selection task, with the grey box representing other techniques that may be developed and added to the system. Each of those modules will be described in greater detail in the following sections.

![Figure 2: AnSelMo’s Architecture.](image)

The system starts by performing a pre-processing step (Section 3.2). From this step results the set of candidate answers\(^3\) together with the snippets or documents that are related with them. These are then passed to one (or more) of the designed techniques. There are three classes of algorithms: Counting (Section 3.3), Lexical Approaches (Section 3.4) – which includes both Word Proximity and Similarity Measures – and Latent Semantic Analysis (Section 3.5). Finally, the scoring is done according to the weights given by each of the modules.

3.2 Pre-Processing

The pre-processing step includes different behaviors, depending on what is intended, which makes this step not totally language independent. In the following paragraphs we detail the possibilities of this step.

As sometimes there is the need to perform query formulation and IR steps (the cases where only the questions along with the candidate answers are given to the system), snippets need to be retrieved for each one of the hypothesis.

\(^3\)Already known apriori.
This is done using two search engines, Bing and Blekko. For each question, distinct queries are envisaged, one for each hypothesis of answer. The process of query formulation involves different formats and filters (the system works with any combination of these): (1) the answer can be quoted or not; (2) the answer terms can appear before or after the question terms (AQ vs. QA format); and (3) different filters can be applied\(^4\): (a) Wh-words–filter, where words such as Where and Who are eliminated; (b) Prepositions–filter, where prepositions like At are removed; (c) To Be–filter, that ignores different forms of the verb to be. These filters are an example of language dependence of the system.

The system also works with local documents (or collection of documents). Each question should be bound to the document where the answer can (or should) be found. Thus, in this situation, the IR does not happen.

### 3.3 Counting

This is one of the simplest modules and is based on the work of Lam et al. [8], described in Section 2. The authors performed experiments with this strategy, accomplishing interesting results. Recall that this strategy is based in the assumption that the correct candidate answer is the one that has the greatest number of hits when querying a search engine. The used queries are obtained as explained in Section 3.2, containing both the candidate answers and question terms.

### 3.4 Lexical Approaches

#### 3.4.1 Word Proximity

Word Proximity strategy is based on the assumption that answers occur close to questions terms. The algorithm calculates the distance between each candidate answers’ terms and the question terms in the neighborhood, that is, it weighs the distances, considering a maximum radius\(^5\), so that documents with too many references to an answer but not to the corresponding question terms are worth less. The algorithm is presented in Algorithm 1 [8]. This way, by consulting a set of documents, one can search for the answers and, in case of finding them, check within the radius for the question terms. The original scoring algorithm is a Linear function. Other variations were developed, in an attempt to minimize the linear decreasing given by the algorithm, and their definition can be consulted in the thesis document.

The index where the answer term was found is \(i\) in the original algorithm. Recall that, when an answer term is found, the algorithm looks for question terms in the surrounding terms, within a given radius; \(j\) represents the index in the window and takes values between \(-radius\) and \(+radius\).

The process is applied for each term present in the candidate answer, which will have a score of 0.0 if no question term is found within the radius.

\[^4\]These are, actually, used throughout the whole system. Henceforth, when referring to filtering in any of the strategies, any combination of these can be applied.

\[^5\]We use the term radius as it was introduced by Lam et al. [8], instead of window. Notice that the size of a window is twice the radius.

---

**Algorithm 1** Pseudocode for a scoring algorithm, giving more weight to answers near question words, within radius words.

```
1: DistanceScore(documentSplit, questWords, ansWords, radius)  
2: score, ansFoundWords = 0  
3: for i = 1 to ||documentSplit|| do  
4:   if documentSPLITed[i] ∈ ansWords then  
5:     ansFoundWords += 1  
6:   for j = (i−radius) to (i+radius) do  
7:     if documentSPLITed[j] ∈ questWords then  
8:       score += (radius −| i − j |)/radius  
9:     end if  
10: end for  
11: end if  
12: end for  
13: if ansFoundWords == 0 then  
14:   return 0  
15: else  
16:   return score/ansFoundWords  
17: end if
```

---

We apply the algorithm to the top documents (passages) or snippets associated.

#### 3.4.2 Similarity Measures

Other approach used by our system is based on similarity measures. For this we use the notion of extents, that is, a passage that includes each term of a given query at least once (as presented by Clarke et al. [5] – see Section 2). The used queries are simply the questions and answers, seen as bags of words, and filtered of stopwords. Thus, we will have an extent for the question (question extent) and many extents as answers (answer extents). As before, those can also be filtered.

This is the first definition we used to create our strategy, and is henceforth referred simply as Extents. However, as an extent must contain all query terms, it may happen that no extent is created because not all terms exist in the text. Thus, if we are not able to identify all query terms in the text, then an extent with only the existing terms will be created. Also, if a given extent has length below a given threshold, we expand it by \(n\) terms at the start and end of the extent; otherwise, the extent would be too short to use the similarity measures. On the other hand, the extents may end up being too large, due to the requirement of meeting such a restrictive constraint. These will damage the similarity measures, because the extents would be too similar.

With this in mind, we developed other strategy to create our extents. This is based on Part of Speech (POS) tagging\(^6\). The idea is to have important words (read nouns and verbs present in the query) to contribute with some weight to the extent. The extent has a score threshold, which, if surpassed, defines the extent. Table 1 shows the scores attributed to each POS. The threshold is defined by two parameters: the tag threshold and the others threshold. The later is set, empirically, to 8.0, while the former is defined in function of the query, and is set to half the total present

\[^6\]Another language dependence point of the system.
in the query. This way we can create extents that contain only parts of the query (thus, reducing their size), but that are still large enough to apply the similarity measures (for example, if the tag threshold is set to 12.0, and we find three Proper Nouns together (3 times 4.0), we still need to find other eight words (8 times 1.0) to complete the extent7). This strategy is called Extents Points.

<table>
<thead>
<tr>
<th>POS Tag</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proper Noun</td>
<td>4.0</td>
</tr>
<tr>
<td>Common Noun</td>
<td>2.5</td>
</tr>
<tr>
<td>Verb</td>
<td>1.5</td>
</tr>
<tr>
<td>Others</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Table 1: Scores for each POS tag.

The extents, from any of the two strategies, are then compared against each other (question extent versus each one of the answer extents). The most widely used similarity measures that do not penalize word order are used in our system: Overlap, Jaccard and Dice, as defined in Equations 1 to 3.

\[
\text{Overlap}(X, Y) = \frac{|X \cap Y|}{\min(|X|, |Y|)}
\]

(1)

\[
\text{Jaccard}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}
\]

(2)

\[
\text{Dice}(X, Y) = 2 \times \frac{|X \cap Y|}{|X| + |Y|}
\]

(3)

We made some modifications to the interpretation given to intersection and union. Usually these similarity measures are used with sets, that is, identical words will collapse into one unique word, either in intersection or union. However, an extent containing only once a given term hardly represents the same passage as another extent containing five times that term. We wanted, thus, to account that. Respecting the similarity measures properties, such as returning values between 0.0 and 1.0, we defined intersection and union in such way that can be applied to these similarity measures:

Intersection The intersection between two extents (two bags of words) is the bag of words that contain all words that co-occur in both extents for each time they co-occur;

Union The union between two extents (two bags of words) is a bag of words containing all words that do not co-occur in both extents plus the words present in the intersection.

An example illustrates better these concepts. Given two extents \( E_1 = \{A, B, C, B\} \) and \( E_2 = \{A, A, B, B\} \), the intersection will be the bag of words \( I = \{A, B, B\} \) and the union the bag of words \( U = \{A^*, B^*, B^{**}, C, A\} \), where \( A^*, B^* \) and \( B^{**} \) are the words from the intersection. As the similarity measures only care about sizes, the union size can easily be computed as \( |U| = |E_1| + |E_2| - |I| \).

3.5 Latent Semantic Analysis

The last technique used by our system is based in LSA. As described in Section 2, LSA computes a matrix \( A \) and then applies SVD to it. After SVD application, both the question and answers are projected over the space defined by matrix \( V^T \). This will result in a set of vectors representing the question and each answer under the space \( V^T \). There are then two approaches that can be used:

A1 In the first approach, we compare the question with each passage present in matrix \( V^T \), selecting the most similar passage to the question. Then we select as correct answer the most similar answer to the selected passage.

A2 In the other approach, we directly compare the question to each answer in the space defined by matrix \( V^T \), selecting as correct answer the one most similar to the question.

Basically, the difference between the two is the indirection point added in the first approach, where an additional comparison is made (the most similar sentence to the question and answer).

To compute similarity in the space defined by matrix \( V^T \), we use the cosine of the angle defined by the vectors representing the passages under comparison (\( x \) and \( y \)):

\[
sim_{\text{cos}}(x, y) = \frac{x \cdot y}{\|x\| \|y\|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}}
\]

(4)

In the definition of the semantic space, we explore different weighting strategies. According to Sahlgren Sahlgren [19], the most effective weighting schemes for small contexts are based on the simple frequency (or even binary), that do not use global weights, such as raw or dampened counts. We explore raw and normalized frequency. As normalization factors we use both the max coordinate value and the sum of the values of the coordinates.

3.6 Scoring

The scoring step is the final one, and receives the scores from each technique. It can combine those, according to given weights, and then sorts the candidate answers decreasingly. The top-most answer is chosen, except in cases where the question is negative, that is, questions that contain the word not. For example, for the question Which of these plays is not written by Shakespeare?, probably the candidate answer with the least score is the one less related with the question and, thus, the correct answer. In these cases, we choose as correct answer the one with the least score. If no answer has a score, then the system will not give an answer.

4 Evaluation

This section presents the application of AnSelMo to each scenario and respective evaluation. We first describe the experimental setup in Section 4.1, followed by the attained results for each technique. We end in Section 4.6 with a discussion of the results.

7Whenever necessary, this expansion is done evenly for both sides of the extent.
4.1 Experimental Setup
First we detail the data set used in our experiments and, then, the evaluation metrics proposed to evaluate the system.

4.1.1 Data Set
In our evaluations we used different data sets, depending on the task being evaluated. As previously stated, we considered three scenarios: WWBM and QA4MRE, which are tasks with multiple-answer questions where one hypothesis is the correct one, and Just.Ask, using as candidate answers the hypothesis extracted by the original system.

‘Who Wants to Be Millionaire?’ Regarding WWBM, we used two corpus in two different languages: Portuguese and English. For the latter we collected a set of questions from some editions of the computer/console game ‘Who Wants to Be a Millionaire?’.

We extracted a subset of 100 questions, randomly picked, without caring about difficulty (in the human point of view the initial ones are the easiest). For the Portuguese set, we used a corpus of 100 questions from the local broadcast, manually transcribed, also randomly chosen among a total of 180 questions.

The description of both corpus can be seen in Tables 2 and 3.

<table>
<thead>
<tr>
<th>Questions</th>
<th>Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>100</td>
</tr>
<tr>
<td>Total words</td>
<td>981</td>
</tr>
<tr>
<td>Unique words</td>
<td>376</td>
</tr>
<tr>
<td>Longest</td>
<td>17</td>
</tr>
<tr>
<td>Shortest</td>
<td>3</td>
</tr>
<tr>
<td>Average</td>
<td>9.81</td>
</tr>
</tbody>
</table>

Table 2: English WWBM corpus characteristics.

<table>
<thead>
<tr>
<th>Questions</th>
<th>Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>100</td>
</tr>
<tr>
<td>Total words</td>
<td>1151</td>
</tr>
<tr>
<td>Unique words</td>
<td>558</td>
</tr>
<tr>
<td>Longest</td>
<td>18</td>
</tr>
<tr>
<td>Shortest</td>
<td>3</td>
</tr>
<tr>
<td>Average</td>
<td>11.51</td>
</tr>
</tbody>
</table>

Table 3: Portuguese WWBM corpus characteristics.

<table>
<thead>
<tr>
<th>Questions</th>
<th>Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>558</td>
</tr>
<tr>
<td>Total words</td>
<td>4143</td>
</tr>
<tr>
<td>Unique words</td>
<td>1274</td>
</tr>
<tr>
<td>Longest</td>
<td>19</td>
</tr>
<tr>
<td>Shortest</td>
<td>3</td>
</tr>
<tr>
<td>Average</td>
<td>7.42</td>
</tr>
</tbody>
</table>

Table 4: QA4MRE corpora characteristics.

<table>
<thead>
<tr>
<th>Questions</th>
<th>Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>558</td>
</tr>
<tr>
<td>Total words</td>
<td>13575</td>
</tr>
<tr>
<td>Unique words</td>
<td>22559</td>
</tr>
<tr>
<td>Longest</td>
<td>12</td>
</tr>
<tr>
<td>Shortest</td>
<td>1</td>
</tr>
<tr>
<td>Average</td>
<td>24.33</td>
</tr>
</tbody>
</table>

Table 5: Just.Ask corpora characteristics.

Question Answering for Machine Reading Evaluation
Considering the QA4MRE challenge, the English 2011 corpus was used. It comprises three topics: “Aids”, “Climate Change” and “Music and Society”. The corpora characteristics can be consulted in Table 4.

Just.Ask Finally, for Just.Ask experiments we used a set of 558 questions. Those questions are a subset of the original corpus used by Just.Ask, and correspond to those for which it is able to extract the correct answer and place it in the top 50 candidates. Thus, at least one or more candidate answers are the correct answer in these questions, having each question at most 50 candidates. The corpus characteristics are in Table 5.

Discussion Regarding the WWBM corpus, we can see that Portuguese is a much more rich and verbose language, using about 200 new words and in average two more words per question. Another important point is the difference between WWBM and QA4MRE corpora. If we measure the number of words by question, we see that the numbers for the QA4MRE corpus are greater than for the English WWBM corpus. This may have to do with the nature of the challenge: QA4MRE is a Machine Reading task and may have more complex questions, requiring knowledge of more words and meanings. On the other hand, these values are smaller than those for the Portuguese WWBM corpus, which corroborates the initial suspicion. On the other hand, QA4MRE answers tend to be longer than the others. Once again, this is strictly related with the task nature. Finally, we can see that Just.Ask corpus is quite similar to the WWBM English corpus.

4.1.2 Evaluation Measures
We evaluate our system according to different metrics. Those aim at better understanding how the system’s techniques perform, being also able to compare the system’s performance in each scenario, although not directly.

QA4MRE task has its own evaluation measure [15]. The measure, called c@1, rewards a system that opts by not answering a question, instead of answering it wrongly. The reward is in function of the number of correct answers. c@1 is defined as follows:

\[
c@1 = \frac{1}{N} \left( n_R + n_U \frac{n_R}{N} \right) ,
\]

where \( N \) is the number of questions, \( n_R \) the number of correct answers and \( n_U \) the number of unanswered questions.

---

8The question set is available at GameFAQs, at http://www.gamefaqs.com/gba/582399-who-wants-to-be-a-millionaire/faqs/37922
The metric is useful to compare the system against others participating in the QA4MRE task. However, this measure will be used in all scenarios. It may be also interesting to evaluate the system only considering the number of correct answers. Thus, accuracy will also be contemplated:

\[
\text{Accuracy} = \frac{n_R}{N},
\]

A third evaluation measure to be applied is Mean Reciprocal Rank (MRR). MRR weights the answers given to each question according to their rank among other candidates to the same question and can be useful to measure how far, overall, the correct answers are from the top:

\[
\text{MRR} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{\text{rank}_i},
\]

with \(\text{rank}_i\) being the rank of the correct answer to the \(i\)-th question, among the answers returned.

### 4.2 Counting

The Counting technique was only applied to WWBM, as only in this scenario an IR step is needed. In QA4MRE the answer should be found in the given document and with Just.Ask the candidate answers are already extracted from a search engine.

Table 6 shows the best results with Counting technique. The columns AQ, Quote and Filter respect to the passage retrieval step (in the query creation): for AQ mode (answer concatenated with the question), the column has true as value, and for QA (question concatenated with the answer), false. Quote represents the quotation of the answer in the query and the filters initials represent the different possible filters presented in Section 3.2.

<table>
<thead>
<tr>
<th>Lang.</th>
<th>Parameterization</th>
<th>Acc.</th>
<th>MRR</th>
<th>C@1</th>
</tr>
</thead>
<tbody>
<tr>
<td>EN</td>
<td>Bing, AQ, Quote, wh_prop, be</td>
<td>0.70</td>
<td>0.802</td>
<td>0.714</td>
</tr>
<tr>
<td></td>
<td>Blekko AQ, Quote, wh_prop, be</td>
<td>0.67</td>
<td>0.802</td>
<td>0.723</td>
</tr>
<tr>
<td>PT</td>
<td>Bing, AQ, Quote, wh_prop, be</td>
<td>0.50</td>
<td>0.679</td>
<td>0.488</td>
</tr>
<tr>
<td></td>
<td>Blekko AQ, Quote, wh_prop, be</td>
<td>0.31</td>
<td>0.565</td>
<td>0.237</td>
</tr>
</tbody>
</table>

Table 6: Best results accomplished with Counting, for WWBM English corpus.

### 4.3 Word Proximity

For WWBM data set, we ran tests with different values for radius and number of passages (respectively 20, 40, 60 and 5, 10, 20). This resulted in 216 different tests. From there we ran more tests, now with values for radius of 3, 4, 5, 6, 10, 30, 50, 70 and 80. In Table 7 are the two best results for both search engines. Although it is not visible in this table, the top results for Bing and Blekko comprise a considerable number of runs with a number of passages equal to 20. This shows that the strategy benefits from analyzing more documents.

Tests for Portuguese corpus used the same parameterizations as English tests, for radius values of 5, 20 and 70. Table 7 shows the accomplished results. It is clear that Blekko is not able to achieve the same results for Portuguese.

For QA4MRE, we use the respective text as document. Results are present in Table 8.

Experiments with Just.Ask used AnSelMo as an answering selection module, together with Just.Ask. In this case, the scores returned by the module are the initial scores of each candidate answer. Table 9 shows the results obtained\(^\text{10}\). Baseline is the best result attained by Just.Ask. Cubic and Cubic Root are other scoring algorithms, and their definition can be consulted in the thesis document.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Radius</th>
<th>Acc.</th>
<th>MRR</th>
<th>Top-3 Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cubic</td>
<td>40</td>
<td>0.612</td>
<td>0.723</td>
<td>0.805</td>
</tr>
<tr>
<td>Cubic_Root</td>
<td>40</td>
<td>0.610</td>
<td>0.721</td>
<td>0.803</td>
</tr>
<tr>
<td>Cubic</td>
<td>30</td>
<td>0.608</td>
<td>0.721</td>
<td>0.805</td>
</tr>
<tr>
<td>Cubic_Root</td>
<td>30</td>
<td>0.608</td>
<td>0.720</td>
<td>0.803</td>
</tr>
<tr>
<td>Linear</td>
<td>30</td>
<td>0.608</td>
<td>0.721</td>
<td>0.801</td>
</tr>
<tr>
<td>Linear</td>
<td>40</td>
<td>0.606</td>
<td>0.719</td>
<td>0.805</td>
</tr>
<tr>
<td>Baseline</td>
<td></td>
<td>0.599</td>
<td>0.714</td>
<td>0.803</td>
</tr>
</tbody>
</table>

Table 9: Best results accomplished with Word Proximity, for Just.Ask corpus, using AnSelMo as an answering selection module.

### 4.4 Similarity Measures

Both Extents and Extents Points were used to WWBM English and Portuguese corpora. The best results for WWBM corpora are shown in Table 10. Blekko was not considered for the Portuguese corpus in these experiments. In Table 11 we present the results for both Extents and Extents Points techniques for QA4MRE corpus.

### 4.5 Latent Semantic Analysis

In this section we report the results attained by using LSA as explained in Section 3.5. The scenarios to which LSA was applied are WWBM and QA4MRE, detailed in the following subsections. Matrix \(A\) is created with terms by documents when applied to WWBM and terms by passages for QA-4MRE\(^\text{11}\).

A1 and A2 are the approaches described in Section 3.5 and Freq., Norm. (Sum), and Norm. (Max) are the different weighting strategies explored.

For WWBM we opted to use the apparent best formula-tions: Bing, AQ, Quoting and BE as filter and Blekko, AQ, Quoting and WH_PROP_BE as filter. Experiments used a different number of passages (1, 5 or 10), meaning that matrix \(A\) will have at most four, twenty or forty documents.

---

\(\text{10}\)Just.Ask calculates Top-3 accuracy instead of c@1.

\(\text{11}\)In Just.Ask scenario we only have snippets and, thus, this technique cannot be applied.
Table 7: Best results accomplished with Word Proximity, for WWBM corpora, using Mean as combination method and Linear as scoring algorithm.

<table>
<thead>
<tr>
<th>Rad.</th>
<th>50</th>
<th>20</th>
<th>30</th>
<th>70</th>
<th>40</th>
<th>60</th>
<th>10</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acc.</td>
<td>0.317</td>
<td>0.317</td>
<td>0.308</td>
<td>0.300</td>
<td>0.300</td>
<td>0.292</td>
<td>0.242</td>
<td>0.233</td>
</tr>
<tr>
<td>MRR</td>
<td>0.523</td>
<td>0.518</td>
<td>0.521</td>
<td>0.517</td>
<td>0.515</td>
<td>0.514</td>
<td>0.452</td>
<td>0.391</td>
</tr>
<tr>
<td>C@1</td>
<td>0.343</td>
<td>0.351</td>
<td>0.334</td>
<td>0.323</td>
<td>0.325</td>
<td>0.314</td>
<td>0.286</td>
<td>0.303</td>
</tr>
</tbody>
</table>

Table 8: Best results accomplished with Word Proximity, for QA4MRE 2011 corpus, using Linear as scoring algorithm, for different radius values.

<table>
<thead>
<tr>
<th>Lang.</th>
<th>Strategy</th>
<th>Parameterization</th>
<th>Acc.</th>
<th>MRR</th>
<th>C@1</th>
<th>Sim.</th>
<th>Passages</th>
</tr>
</thead>
<tbody>
<tr>
<td>EN</td>
<td>Extents</td>
<td>Bing, QA, prop_be</td>
<td>0.53</td>
<td>0.705</td>
<td>0.541</td>
<td>Dice</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Blekko, AQ, Quote, wh_prop_be</td>
<td>0.67</td>
<td>0.753</td>
<td>0.764</td>
<td>Dice</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Extents Points</td>
<td>Bing, QA, Quote, prop_be</td>
<td>0.72</td>
<td>0.817</td>
<td>0.763</td>
<td>Dice</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Blekko, AQ, Quote, wh_prop_be</td>
<td>0.60</td>
<td>0.706</td>
<td>0.690</td>
<td>Dice</td>
<td>20</td>
</tr>
<tr>
<td>PT</td>
<td>Extents</td>
<td>Bing, QA, Quote, prop_be</td>
<td>0.40</td>
<td>0.575</td>
<td>0.448</td>
<td>Jaccard</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bing, QA, Quote, prop_be</td>
<td>0.44</td>
<td>0.577</td>
<td>0.519</td>
<td>Dice</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 10: Best results accomplished with Similarity Measures, for WWBM corpora, using different similarity measures and both Extents and Extents Points.

<table>
<thead>
<tr>
<th>Sim.</th>
<th>Overlap</th>
<th>Jaccard</th>
<th>Dice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acc.</td>
<td>0.242</td>
<td>0.267</td>
<td>0.267</td>
</tr>
<tr>
<td>MRR</td>
<td>0.502</td>
<td>0.532</td>
<td>0.532</td>
</tr>
<tr>
<td>C@1</td>
<td>0.248</td>
<td>0.273</td>
<td>0.273</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sim.</th>
<th>Overlap</th>
<th>Jaccard</th>
<th>Dice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acc.</td>
<td>0.192</td>
<td>0.233</td>
<td>0.233</td>
</tr>
<tr>
<td>MRR</td>
<td>0.386</td>
<td>0.402</td>
<td>0.402</td>
</tr>
<tr>
<td>C@1</td>
<td>0.248</td>
<td>0.301</td>
<td>0.301</td>
</tr>
</tbody>
</table>

Table 11: Results accomplished with Similarity Measures, for QA4MRE 2011 corpus, using different similarity measures and both Extents and Extents Points.

<table>
<thead>
<tr>
<th>App.</th>
<th>A1</th>
<th>A2</th>
<th>A1</th>
<th>A2</th>
<th>A1</th>
<th>A2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freq.</td>
<td>0.192</td>
<td>0.258</td>
<td>0.175</td>
<td>0.267</td>
<td>0.192</td>
<td>0.242</td>
</tr>
<tr>
<td>Norm. (Sum)</td>
<td>0.212</td>
<td>0.286</td>
<td>0.194</td>
<td>0.296</td>
<td>0.212</td>
<td>0.268</td>
</tr>
<tr>
<td>Norm. (Max)</td>
<td>0.192</td>
<td>0.242</td>
<td>0.192</td>
<td>0.242</td>
<td>0.212</td>
<td>0.268</td>
</tr>
</tbody>
</table>

Table 13: Results accomplished with LSA, for QA4MRE 2011 corpus.

4.6 Discussion

In the chapter we presented the different techniques that AnSelMo incorporates, as well as their results in three different scenarios. Results are different between them, and not standardized. In fact, the best parametrization for one scenario, in a given technique, is not the same for another scenario.

Counting technique performs as well as most techniques, but it is probably not accurate for a real QA system, as in the scenario it was applied (WWBM) candidates are not related and, thus, corresponding queries may be sufficient to choose the correct answer. On the other hand, LSA obtained the worst results, even in a scenario like QA4MRE. In WWBM we used a set of documents related with each of the candidate answers, so probably there are some latent topics that are shared by all answers, not allowing the sys-

---

12Unfortunately, how the framework actually works, it is not possible to calculate the MRR results.
tem to distinguished them. However, in QA4MRE, we only used one document – the test document. The poor results may be related with the task nature, and, for this reason, other features to fill matrix $A$ should be studied, such as $tf.idf$ scores.

Similarity Measures had some interesting results, namely Extents Points, as this technique can improve significantly Extents’ Bing results. However, a deeper analysis must be done, as it is not clear their impact when compared with Word Proximity.

When compared with state of the art, results obtained for WWBM are better. However, our results cannot be compared straightforward with results obtained in the beginning of this century (around 70%-75% accuracy), as we are using different search engines and different corpora.

Regarding QA4MRE, results are lower than the 2011 winning system, but they surpass almost all other competing systems.

For WWBM Portuguese corpus, results are lower than for English and are quite similar among the studied techniques, although with highlight for Word Proximity, once again. We believe, however, that these lower results are strongly related with the less information that can be found on the web.

Finally, Just.Ask results are promising. The best conclusion we can draw from the obtained results is that our techniques can, in fact, help a QA system improve its performance, by pushing more often the correct answer to the top.

## 5 Conclusions and Future work

In this thesis we studied the answer selection problem, that aims at selecting one or more candidate answers as correct answers. We propose an approach based on context: given a candidate answer, we account the text where it was extracted.

We developed AnSelMo, an answering selection module, based on some previous state of the art techniques. Besides Counting, which chooses the answer only based on the number of search engine hits, AnSelMo implements two different approaches: Lexical approaches and a Semantic Space-based approach. The first includes Word Proximity, an algorithm that weights the distance between question and answer terms, and Similarity Measures, which compares the extents found, that is, the passages that represent the question and the answers, by containing some terms from them. Latent Semantic Analysis is the Semantic Space technique used, and tries to capture latent topics from the documents. These documents are related with the candidate answers.

We used AnSelMo in three different scenarios: WWBM, QA4MRE and Just.Ask. The first two contain questions with a fixed number of candidate answers, non-related, where one is the correct answer. The later is a QA system, where questions have a different number of candidates that can be related. Results showed that the techniques can surpass state of the art results for WWBM scenario, equalize most 2011 QA4MRE systems and, using AnSelMo as an module in Just.Ask, improve the QA system results.

Although AnSelMo accomplished promising results in all scenarios, there is still much room for improvement. Some possible extensions are the following: (1) different Information Sources can be used. In this thesis we only used Bing and Blekko as search engines, and FreeLing POS tagger. However, the usage of WordNet and a stemmer/lemmatizer would allow the system to identify synonym words or conjugations of verbs as the same word; (2) other Semantic Space models could be applied, as pLSA or LDA, as they use a probabilist model, allowing words to belong to more than a topic, even if not explicit in the initial documents, by attributing a small probability of the word belonging to such topics. Also, different strategies to create matrix $A$ could be developed, as using $tf.idf$ scores, collapse synonyms, etc.; (3) the presented techniques can be more adequate to some questions than others, and may, thus, be able to answer to a different set of questions. The different ordinations could be merged into a final ordination, increasing this way the number of correct answers. Techniques that combine the returned scores should be studied, such as self-learned weights or Learn to Rank algorithms; and (4) the Extents and Extents Points techniques were both used to apply Similarity Measures, but they could also be used to identify the related snippets and, then, be used in Word Proximity technique. Also, other similarity measures could be used, including those that care about order, such as cosine similarity.

### References

João Pedro Carlos Gomes da Silva. Qa+ml@wikipedia&google, November 2009.


S.K. Lam, D.M. Pennock, D. Cosley, and S Lawrence. 1 billion pages = 1 million dollars? mining the web to play "who wants to be a millionaire?". In Uncertainty in Artificial Intelligence (UAI2003), pages 337–345, Acapulco, Mexico, 2003.


