Query Reformulation and Answer Retrieval
The task of automated candidate answer extraction from the Web requires pre-fetching of documents before actual answer extraction. Pre-fetching of documents is performed by posing a reformulated query from the user question to the information retrieval engine. This chapter describes the task of question reformulation to query for retrieving information either from QA corpus or Web.

5.1 Introduction

Fetching most relevant information from the vast collection, one has to ensure proper reformulation of the query before posing to the information retrieval interface. Appropriate query reformulation increases the possibility of inclusion of the most relevant documents or answers in the retrieved list. For query reformulation one often needs to use the applications from deep or shallow NLP (natural language processing) techniques, pre-stored knowledge information, language modeling and pattern mining. Thus, query reformulation has received attention from these areas and become a testing ground how all of these can be integrated to bring forth an effective approach. More recently, many of the question answering system either of open or restricted domain have used external information extraction, such as surface text patterns for answer finding. We have also tried to explore these surface patterns for our query reformulation phase with slight modification.

Our assumption is that, in the text-based question answering system, the structural arrangements of the parts-of-speech can also be used efficiently to reformulate queries instead of words and special user defined tokens. We have considered sequential patterns composed of POS (part-of-speech) tags. For instance, the question ‘Who is Chairperson of U.G.C?’ has POS pattern as ‘WDT + VBD + NN + IN + NN’. The same POS pattern can be used to frame other similar questions like ‘Who is X of XYZ University’. Consideration of more generalized sequence of POS patterns instead of text patterns decreases the amount of human effort and count of pre-stored patterns in the corpus and increases the likelihood of fetching more relevant information. Additionally, we have also adapted the concept of structure-based query reformulation proposed by Lioma et
al. (Lioma et al., 2008) for the questions whose pattern is not found in pattern corpus. The structure-based reformulation also relies on syntactical information, but it gives more flexibility for query reformulation than pattern matching approach. The structure-based query reformulation unveils the most probable shallow syntactic structures which are likely to occur or co-occur. We believe that shallow syntactic information will also improve the likelihood of fetching more relevant documents even after erroneous question classification.

For structure-based query reformulation, the syntactical evidence which represents general language needs to be extracted from large-scaled language samples. However, instead of large language samples, we have used question pattern corpus as structural evidence seeing that pattern corpus itself contains more feasible, refined and frequently used text segments within the higher education domain. The pattern corpus can be considered as much reliable resource for syntactical evidence to be used for query reformulation. Meanwhile, our approach to consider question pattern corpus as structural evidence is different from using the same as pattern matching clue. The task of pattern matching focuses on determining the degree to which a question pattern is identical to each candidate pattern while to be used as syntactical evidence, the whole question pattern corpus is processed and filtered to find the most frequent syntactic structure of said length. The syntactic structure that is to be utilized for reformulation can be considered as POS block and arbitrarily set to any length but in our proposed work, we have experimented with three, four and five tokens respectively.

In this regard, our proposed technique first attempts to reformulate the query using the simple and efficient approach of pattern matching and failing to that acquires for structure-based query reformulation to add robustness to our reformulation effort.

5.2 Existing Work

Use of surface patterns for query analysis is not new to the information retrieval field. Allan and Ragahavan (Allan et al., 2002) applied part-of-speech (POS) tagging for query disambiguation. Common patterns of POS tags have been extracted and frequently occurring patterns of query terms across the corpus have been identified to formulate natural language question for query disambiguation. Park and Croft (Park et al., 2010)
used parts-of-speech to assist in identifying the focus of the query. Dumais et al. (Dumais et al., 2002), had utilized redundancy available in large corpora as an important resource to simplify the query rewrites. These query rewrites are further used to support answer mining from extracted snippets through backend retrieval engine. The proposed system has incorporated the n-gram extraction technique and its performance enhances with increasing number of available matching passages. Kaisser et al. (Kaisser et al., 2004) have used manually learned patterns reformulations. Arephrasing algorithm has been developed based on linguistic patterns that describe the structure of questions and candidate sentences and where precisely to find the answer in the candidate sentences. Question answering task proposed by Cucerzan et al. (Cucerzan et al., 2005) used complementary models for structured and unstructured Web content. However, for unstructured content, the Web-based system used snippet pattern matching and generic answer type matching using Web counts.

The QA system presented by Hildebrandt et al. (Hildebrandt et al., 2004) has implemented a multi-strategy approach for answering definition questions. The system is built upon an offline constructed database with surface patterns, a Web-based dictionary, and an off-the-shelf document retriever. The surface patterns used here are operated both at word level and part-of-speech level. In total, eleven surface patterns have been used over the entire AQUAINT corpus. Application of these surface patterns over AQUAINT corpus yield the target, pattern type, nugget and source sentences which are later on stored in relational database. In addition to surface patterns mechanism, a Web-based dictionary has also been used for answering definition question. Failing to these two strategies, the system employs traditional document retrieval strategy to extract relevant nuggets. In addition, the authors have also analyzed the difficulty of evaluating definition questions and the inability of present metrics to accurately capture the information needs of real-world users.

In the TREC 2004, many of the participating systems have opted for surface pattern methods in question answer processing. The representation of QA-LaSIE in particular track has used a multi-strategy approach which uses surface matching text patterns as the primary mechanism for extracting answer for factoid question. The system only moves to next approach of semantic type extraction and WordNet extraction when fails to retrieve
same through pattern matching. Another system which has participated in TREC 2004 is ILQUA. The ILQUA (Wu et al., 2005) has also used surface text pattern matching along with n-gram proximity search and syntactic dependency matching for answer extraction of list and factoid question. The patterns employed in ILQUA are automatically generated by a supervised learning system and expressed in regular expressions form which can handle up to 4 question terms. The questions whose answer cannot be retrieved by surface text pattern matching are thereafter subjected to a combined method of n-gram proximity search and syntactic dependency matching. The overall accuracy achieved by ILQUA for factoid question is 30.9%. However, the representation of ILQUA (Wu et al., 2007) in TREC-2007(Dang et al., 2007) has integrated relative word analysis along with the patterns and semantic features to answer “Other” questions in order to improve the precision.

In contrast to the hard patterns, Cui et al. (Cui et al., 2004a) have demonstrated a flexible representation of patterns which can effectively accommodate the diversity of definition sentence structure exhibited in the news and named them as ‘soft patterns’. Soft patterns consider each slot as a vector of words and syntactic classes along with their distributions, instead of generalizing specific instances to simplify rule. A pseudo relevance feedback has been also used to automatically label sentences for use in soft pattern generation. The approach has shown a performance improvement by 14% for definitional question answering task in TREC 2003. Later Cui et al. (Cui et al., 2004b) have combined definition sentences retrieved from soft pattern approach to the definitions downloaded from external resources and used them to answer list and factoid questions. However, such a definition sentence base restricts recall in passage retrieval. Enhancing the concept of soft patterns, Cui et al. (Cui et al., 2007) have proposed two generic soft pattern models, one based on a bigram language model and the other on the PHMM to identify definition sentences. The experimental results showed that both models significantly outperform the system versions using hard matching patterns. Although, PHMM is more capable of dealing with gaps in pattern matching caused by language variations by performing insertion and deletion editing operations. Though, the authors have restricted their approach of soft pattern matching to definitional question answering only but suggested its application to factoid question answering too.
Similarly, the use of shallow syntactical information to enhance the performance of QA system is not a new concept. Many research works have been carried out to use syntactical information in the form of pseudo syntactical rules for question classification and other different modules of question answering system. Query reformulation has been done using various mechanisms in earlier works. Agichtein et al. (Agichtein et al., 2001) used regular expressions to generate queries while Radev et al. (Radev et al., 2001) utilized a classifier that decides which operator is best applied to a given question to generate the best query paraphrase. Brill et al. (Brill et al., 2002) reformulated query by generating rewrites as on simple string-based manipulations. They used a small percentage of rewrites, in order to determine the possible parts-of-speech of a word as well as its morphological variants instead of using a part-of-speech tagger itself. Although the rewrite rules and associated weights are generated manually for the system, but they have not denied the possibility of learning query to answer reformulations and their weights automatically.

The Korea University Question Answering System at TREC 2004 (Han et al., 2004) has also used syntactic patterns that are used for extracting the answer candidates from the sentences for ‘other’ questions. These patterns are applied to sentences to extract answer phrases. Further enhancing their research work Han et al. (Han et al., 2005) proposed a definitional question answering system based on linguistic information and definition terminology-based ranking. Phrases were extracted using the same concept of syntactic pattern to reduce redundancy based on the lexical gap and semantic matching as discussed in their previous work. Evidence like external definitions and definition terminologies has been used to rank the phrases. Definition terminology scores are actually provided to assist the completion of incomplete external definitions.

Syntactic analysis of the question and passages was also employed by the IBM DeepQA project in order to validate candidate answers (Murdock et al., 2012). Carmel et al. (Carmel et al., 2014) have analyzed the effect of syntactic analysis on term weighting for queries in Community Question Answering (cQA) site. They have proposed a term weighting method which uses syntactic information for each query term occurrence in the document instead of statistical term occurrence methods. The authors have performed manual evaluation over a large log of Web queries of Yahoo Answer site and showed that
syntactic analysis such as part-of-speech tagging and dependency parsing can be used as a complimentary method to statistical term weighting while query formulation. In the similar spirit, another research work in cQA proposed by Figueroa et al. (Figueroa et al., 2014) that has identified effective paraphrase for questions prompted by community members. These paraphrases can offer more effective suggestions to the user by detecting similar past questions and relevant answers. The authors had carried out a number of different large-scale experiments using logs from Yahoo! Search and Yahoo! Answers, and illustrate that the subjective and objective nature of cQA questions affect the recall and ranking of past answers when fine-grained category information are taken into account. In the earlier work presented by the authors (Figueroa et al., 2013), a method for ranking query paraphrase has been implemented in cQA which automatically extracts a corpus of paraphrases of queries and questions using the query-question click history for the query logs from Yahoo! Search and Yahoo! Answers. These paraphrases are automatically ranked in terms of recall and MRR, and thus used for structuring several learning to rank models, i.e. general and specific to question-types, and according to both metrics. The work also concluded that question-type models are more effective than general ranking models.

5.3 Proposed Query Reformulation Method

Question reformulation is considered as one of the most important aspects of question processing in automated question answering. More accurate is the reformulation of question, more relevant is the retrieved result. We, therefore, put our effort to present a reformulation technique which not only compatible with the underlying architecture of our proposed QA system but also imparts robustness to our approach. Our QA system utilizes both question answer corpus and Web as the information resource, hence require a reformulation technique which not simply interacts with question answer corpus but also have features to filter irrelevant information from the Web. Therefore, we came forward with an integrated question reformulation approach which primarily adapt simple pattern matching to formulate the question and failing to that acquires structure based reformulation. Working mechanisms for both the approaches have been discussed in upcoming sections.
5.3.1 Reformulation using Pattern Matching and Filtering

The idea of pattern matching to reformulate query is tailored to bring simplicity along with efficiency. However, implementation of pattern matching requires a resource to be used as the reference for question reformulation. Therefore, we have put our effort to build a manually POS tagged question corpus for reformulation purpose. The corpus is composed of 211 distinct question patterns. The structure of question pattern corpus is shown in figure 1. Each question pattern defines question, tag sequence, question class, focus word and reformulated query. The tags other than <RQuery> are not of our concern for this chapter as they have been incorporated for focus word identification and question classification. We organized question patterns according to the wh-words (what, who, how, where, when, which, why). Penntree Bank tag set Marcus et al. (Marcus et al., 1993) has been used for tagging the question patterns. To design the training set of patterns, we have gathered 549 questions from various online higher education portals and forums, FAQs from various universities’ Websites and other similar resources. The collected questions are then processed through POS tagger to generate POS patterns. Rudimentary chunking, such as marking the boundaries of the noun phrase, is performed by grouping the tags on the basis of their part-of-speech. Distinct patterns are then selected to be included in the question pattern corpus. Meanwhile, it is also ensured that at least one type of question pattern must be crafted for each question from the collection.

Whenever, a new question is introduced to the QA system by the user, it requires implementation of POS tagger to generate a sequence of POS tag. The tagged user question pattern is then chunked to identify noun phrases and presented for pattern matching measurement. The degree of pattern matching is calculated as tag order similarity based on the formula demonstrated by Li et al. (Li et al., 2006) which has proven an efficient metric for determining word order similarity. The formula for determining the coefficient pattern matching as tag order similarity can be demonstrated as follows:

$$S_r = 1 - \frac{|p_q - p_c|}{|p_q + p_c|} \quad (5.1)$$

Where \(p_q\) is the tag order vector for question pattern and \(p_c\) is the tag order vector for candidate pattern from question pattern corpus. The tag order vector gives the basic
structural information for the question and candidate pattern. A unique index number is assigned to each tag according to the order number that the tag occurs in the pattern. The value of $S_r$ lies between 0 and 1 and reaches maximum i.e., 1 when there is no tag order difference. The patterns from corpus which have higher final matching score than a threshold $\lambda$ are candidate patterns for the question pattern and the pattern having highest matching score $S_r$ is the final matched pattern.

Assuming that user question $Q$ has the pattern as $QP$ while that of candidate pattern from corpus as $CP$.

$Q: \text{What grade is given by NAAC to the BBA University?}$

$QP: \text{What_WP grade_NN is_VBZ given_VBN by_IN NAAC_NNP to_TO BBA_NNP University?}_\text{NNP}$

$CP: \text{What_WP courses_NNS are_VBP offered_VBN to_TO students_NNS in_IN BBAU?}_\text{NNP}$

For our processing, we will only consider tags for $QP$ and $CP$. Therefore, we can now consider join set $L = \{WP, NN, VBZ, VBN, IN, NNP, TO, NNS, VBP\}$. The tag order vector for $QP$, i.e., $p_q$ is constructed based on the existence and tag to tag similarity between joint set $L$ and $QP$. Therefore, we will have $p_q = (1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 2 \ 3)$, the value 2 is assigned two times in the vector because the tag $NNS$ being tags for common nouns strongly similar with $NN$, which is at the second position in $QP$. Similarly, we will get $p_c = (1 \ 2 \ 3 \ 4 \ 7 \ 8 \ 5 \ 2 \ 3)$. Using the tag similarity in equation (1), we will get $S_r(p_q, p_c) = 0.864$. The formula proposed by Li et al. (Li et al., 2006) is an efficient metric for measuring tag order similarity which determines the normalized difference of tag order.

### 5.3.2 Syntactic Structure based Query Reformulation

In automated QA, apart from question classification, pattern matching as shallow syntactic information can also be used to reformulate the queries and increase the likelihood of fetching more relevant documents. We propose text structure-based query reformulation as a technique that aims to automatically enhance the performance of a QA system on the basis of shallow syntactic evidence.
We used tag-based blocks because we believe that syntax plays an implicit role in communicating meaning. POS tag information may model the structure of language by showing which shallow syntactic structures are more likely to occur or co-occur.

1. Syntactic evidence from pattern corpus and question answer corpus.
2. Represent the question syntactically.
3. Reformulate the question using structural evidence fetched from the corpus.

Table 5.1 represents the top 5 most frequent POS blocks for different k values which are to be used as syntactical evidence for structural query reformulation. These blocks aid in reformulating queries that are further used to retrieve candidate answer or document list.

### Step 1. Extracting shallow syntactic evidence from Corpora

The shallow syntactic evidence has been drawn from the question pattern corpus and from question answer corpus as well. The question pattern corpus we are employing at question analysis phase already contains POS tagged pattern (Singh et al., 2014). Likewise, the QA corpus is collection of question-answer pairs where each pair is annotated with a POS tagged snippet paragraph containing the respective answer. These paragraphs are articulated in the QA corpus to aid in automated pattern learning. These corpora have been used as samples to provide syntactic evidence. Depending on the application to Question Answering System in Higher education domain, we have fixed the length of the POS blocks to three and five. These overlapping blocks are analyzed to extract the evidence as implemented by Lioma et al. (Lioma et al., 2008). The POS blocks extracted at this stage are used for further processing at next stage.

<table>
<thead>
<tr>
<th>POS blocks</th>
<th>Test sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>NN IN NN</td>
<td>Question Pattern corpus (k=3)</td>
</tr>
<tr>
<td>VBZ IN NN</td>
<td></td>
</tr>
<tr>
<td>IN NN IN</td>
<td></td>
</tr>
<tr>
<td>NN VBD NN</td>
<td></td>
</tr>
<tr>
<td>VBZ NN IN</td>
<td></td>
</tr>
<tr>
<td>VBZ NN IN NN IN</td>
<td>Question Pattern corpus (k=5)</td>
</tr>
<tr>
<td>NN VB IN NN IN</td>
<td></td>
</tr>
<tr>
<td>NN VB NN IN NN</td>
<td></td>
</tr>
<tr>
<td>NN IN NN IN NN</td>
<td></td>
</tr>
</tbody>
</table>
Step 2. Syntactic representation of Question

The user question is pos tagged and chunked before pattern matching analysis. Therefore, from the shallow syntactic illustration of the question, top t most probable blocks from the corpora samples are extracted. The number of the most probable POS blocks t from the samples is chosen empirically. Examples are illustrated to render better understanding of this step. First, we give the illustration for k=3 followed by k=5 in Figure 5.1 and Figure 5.2 respectively, where k is the number of tokens in POS block.

The most frequent POS blocks filtered from Step 1 through samples are mapped to the POS blocks obtained from the given examples. The selected sequence for the number of tokens, k=3 is <3, 5, 7>. Similarly, for k=5, the selected sequence is <4, 5, 7>.

**Q:** What is the recognition status of various programs for the Universities through distance mode?

**PQ:**
```
WDT VBZ DT JJ NN IN JJ NNS IN DT NNS IN JJ NN
```

**CQ:**
```
WDT VBZ NN IN NNS IN NNS IN NNS IN NN
```

1. \[What \text{ the recognition status}\]
2. \[is \text{ the recognition status of}\]
3. \[the recognition status of various programs\]
4. \[of various programs of\]
5. \[various programs for the Universities\]
6. \[for the Universities through\]
7. \[the Universities through distance mode\]

**Figure 5.1.** POS blocks Illustration for three tokens
Chapter 5 Query reformulation and Answer Retrieval

Q: Where can I find an application for Ph.D. program in Economics department of BBAU?

PQ: WDT MD PRP VB DT NN IN NNP NNS IN NNP

can I find an application

CQ: WDT MD PRP VB NN IN NNS IN NNP

Where can I find an application

can I find an application for

I find an application for Ph.D. program

find an application for Ph.D. program in

an application for Ph.D. program in Economics department

for Ph.D. program in Economics department of

Ph.D. program in Economics department of BBAU

Figure 5.2. POS blocks Illustration for five tokens

Step 3. Reformulation of question using structural evidence

For the reformulation of a question, we would only consider the single occurrence of the tokens from the selected sequences of POS blocks to avoid redundancy and reduce the vague representation of the question. The integration of non-overlapped sequences yields our new query for the QA system. We will first try to select two most frequent distinct POS sequence. Therefore, our reformulated query without overlap, for k=3 is ‘the recognition status of various programs for the universities’ and for k=5 is ‘find an application for Ph.D. program in Economics department’.

5.4 Query Similarity Evaluation and Answer/Document Retrieval

This segment of our proposed scheme is dedicated to the answer candidate retrieval or the document retrieval of the reformulated query from question answer corpus and search engine respectively.

The question answer (QA) pair corpus actually stores the question in the form of reformulated query and their corresponding answer. The notion of working with query instead of question has been taken to reduce the redundancy in the corpus for different representation of the same question. For QA corpus, it is assumed that all accumulated questions were posted with somewhat syntactic pattern merely represented as query along
with their corresponding answer. These query-answer pairs within corpus are classified
into eight different sub-domains within the higher education domain viz., History, About,
Admission and Scholarships, Academics, Examination, Events and Finance to provide
efficient searching in the corpus (Singh et al., 2015b). Identification of a relevant sub-
domain inside the corpus requires implementation of a simple algorithm based on domain
specific criterion proposed in the work. After discovering the relevant sub-domain, we
extract candidate question set \((qc_1, qc_2, qc_3, \ldots, qc_n)\) by calculating the similarity of user
question \(q\) with the help of following formula presented by Li et al. (Li et al., 2006) for
measuring word order similarity as follows:

\[
W_r = 1 - \frac{|u_q - u_c|}{u_q + u_c}
\]  

(5.2)

Where \(u_q\) is the word order vector for user reformulated query \(q\) and \(u_c\) is the word order
vector for candidate query from question-answer pair corpus. The value of \(W_r\) lies
between 0 and 1 and reaches maximum i.e., 1 when there is no word order difference.
The queries from corpus which have higher final matching score than a threshold \(\mu\) are
candidate queries and that having highest matching score \(W_r\) is the final matched query.
The corresponding answer for the highest matching query is considered as the final
answer for the user reformulated query.

However, if no candidate query shows the similarity measure greater than threshold \(\mu\)
with respect to the user’s reformulated query, the later is forwarded to search engine for
retrieving relevant documents. These documents will be further analyzed to retrieve
answer of the user question. But for now, we are only interested in analyzing the
relevancy of the returned document list against the reformulated query to evaluate the
effectiveness of our proposed integrated query reformulation approach. In proposed
article, we have checked the document relevancy against a reformulated query with the
help of tf – idf and Okapi BM25 both. The overall process of query reformulation using
integrated approach has been shown in Figure 5.3.
Motivated by these reformulation mechanisms and similarity measurements, we have presented the integrated question reformulation algorithm as in Figure 5.4. In this algorithm, we first try to reformulate question with pattern matching. When pattern matching fails to produce any candidate pattern, the algorithm moves forward for syntactic structure based query reformulation. Currently, for structural reformulation, we opted for a choice of POS block length on behalf of the main verb which is later on justified with the help of experimental analysis.

**Algorithm** Question Reformulation

**Input:** Question q

**Output:** Reformulated query q_r

**Begin**

1. For each question q

---

86
2. Apply a part-of-speech tagger and a noun phrase chunker to q.
3. Compute similarity coefficient between tagged question $q_t$ and patterns contained in question pattern corpus.
4. If similarity coefficient $S_r$ is beyond the threshold value $\lambda$.
5. Pattern with highest similarity coefficient is the candidate pattern for the question.
6. Fetch the reformulated query pattern and accordingly recreate the query $q_r$ by inserting focus word and other primitives at appropriate position, return $q_r$.
7. Otherwise, apply structural query reformulation.
8. If tagged and chunked question $q_t$ has a main verb, then select length of POS block i.e., $k = 5$ otherwise $k = 3$.
9. Frame $q_t$ according to value of $k$.
10. Reformulate query $q_r$ using structural (syntactic) evidence drawn from question pattern corpus.
11. return $q_r$

**End**

**Figure 5.4. Query Reformulation Algorithm**

After the proper query reformulation, the task of information retrieval has to be performed. The retrieved information could be a list of answers if filtered from question answer corpus or a list of documents if posed on the search engine. Hence, we first seek to identify relevant answer list from the QA corpus. Identification of candidate answers require selection of suitable candidate queries from QA corpus which are selected according to the two-way classification mechanism explained by Singh et al. (2015b). The classification mechanism executes in two phases, i.e., primary and secondary. The primary stage belongs to question classification and focus word detection task. The second phase actually is the decisive phase for selection of candidate queries from QA corpus depending on the domain based constraints. After selecting candidate queries, similarity coefficient is calculated according to the algorithm presented in figure 4. However, if the similarity coefficient for no candidate queries lies above threshold value $\mu$, the user query is forwarded to the search engine for document extraction. The search engine extracts and thereafter ranks matching documents according to the relevance for
the query based on ranking function used in our proposed method, i.e., tf-idf or Okapi BM25.

Algorithm Answer/ Document retrieval

**Input:** Reformulated query \( q_r \)

**Output:** Candidate answer list \( C_a = \{a_i, \ldots, a_n\} \) / documents \( C_d = \{d_i, \ldots, d_n\} \)

**Begin**

1. For each reformulated query \( q_r \)
2. Select candidate queries \( Q_i \) from question answer corpus.
3. Compute similarity coefficient \( W_i \) between reformulated query \( q_r \) and candidate query \( Q_i \), i.e., \( W_i = \text{Sim}(q_r, Q_i) \)
4. If \( W_i \geq \mu \), Query \( Q_i \) is the candidate query.
5. Fetch the corresponding candidate answer.
6. Repeat Step 3 to 6 for each candidate query
7. Form a list \( C_a \) for top \( k \) answers having higher \( W_i \) value.
8. Else, pose the reformulated query \( q_r \) to the search engine API.
9. Retrieve top \( n \) candidate documents list \( C_d \) based on ranking function.

**End**

**Figure 5.5.** Information Retrieval algorithm

### 5.5 Query Reformulation Evaluation

We validated the plausibility and effectiveness of the proposed query reformulation method by carrying out our experiments on a question collection of 1250 questions. Here, we did use of the different variant of same question to check the effectiveness of our proposed technique. The questions in this data are related to higher education domain. For example, questions about placements, conferences, scholarships etc. These questions cover each class of QA corpus as defined by Singh et al. (2015b).

#### 5.5.1 Evaluation Measure

There are a number of evaluation measures that can be used to compare the performance of the different retrieval techniques. Each measure highlights a different aspect, for
instance, it is common to use non-interpolated average precision, also referred to as mean average precision (MAP), in ad hoc retrieval as it is an important indicator of the quality of the document collection retrieved as well as of similarity matching score formula. Given a query \( q \), its set of relevant documents \( REL_q \) and a ranking of documents resulting from the retrieval process, average precision of an individual query is defined as:

\[
avg\_prec(q) = \frac{\sum_{d \epsilon REL_q} \left[ \left[ \frac{\text{rank}(d) \leq \text{rank}(d)}{\text{rank}(d)} \right] \right]}{\text{REL}_q}
\]  

The mean average precision is then simply the mean of all individual average precisions.

We have conducted our experiment in two phases for the query reformulation on the test collection. The first test shows the impact of query reformulation for question-answer pair corpus for both pattern matching and structural query reformulation (SQR). We have conducted a baseline experiment which has used POS blocks of four tokens without chunking as carried out by Lioma et al. (Lioma et al., 2008). We have evaluated the same set of questions after chunking for the POS blocks having three, four and five tokens. Table 5.2 shows the tabulation of MAP score for pattern matching and SQR on QA corpus. In proposed work we are not interested in the process of fetching exact answer as our major concern is to evaluate the performance of query reformulation against relevant information extraction which may either be list of documents or answers. Therefore, we have presented our analysis based on precision and similarity based formulae to ensure the likelihood of fetching more relevant query answer pair rather than the exact answer. The questions for which, we are not able to retrieve answers through QA corpus, are transferred to the search engine for filtering relevant documents against the reformulated query. In quest of relevant information we are relying on top 10 results from question answer corpus, however, number of relevant results is increased up to 50 in case of Web owing to its large size. The MAP score for search engine retrieval is shown in Table 5.3 for both pattern matching and SQR. The table also tabulated the difference in performance as compared to baseline for POS blocks with three, four and five tokens with chunking.
### Chapter 5 Query reformulation and Answer Retrieval

<table>
<thead>
<tr>
<th>QC class</th>
<th>Pattern matching</th>
<th>Tf-idf</th>
<th>SQR</th>
<th>Okapi BM25</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Baseline (k=4 without chunking)</td>
<td>K=3</td>
<td>Δ%</td>
</tr>
<tr>
<td>History</td>
<td>0.529</td>
<td>0.334</td>
<td>0.365</td>
<td>+9.3</td>
</tr>
<tr>
<td>Who’s Who</td>
<td>0.543</td>
<td>0.229</td>
<td>0.359</td>
<td>+9.1</td>
</tr>
<tr>
<td>Events</td>
<td>0.489</td>
<td>0.321</td>
<td>0.349</td>
<td>+8.7</td>
</tr>
<tr>
<td>Finance &amp; Scholarship</td>
<td>0.521</td>
<td>0.331</td>
<td>0.363</td>
<td>+9.7</td>
</tr>
<tr>
<td>Examination &amp; Admission</td>
<td>0.498</td>
<td>0.325</td>
<td>0.345</td>
<td>+6.2</td>
</tr>
<tr>
<td>Establishment</td>
<td>0.474</td>
<td>0.311</td>
<td>0.318</td>
<td>+2.2</td>
</tr>
<tr>
<td>Academics</td>
<td>0.489</td>
<td>0.323</td>
<td>0.334</td>
<td>+3.4</td>
</tr>
<tr>
<td>About</td>
<td>0.481</td>
<td>0.314</td>
<td>0.321</td>
<td>+2.3</td>
</tr>
<tr>
<td>Total</td>
<td>0.503</td>
<td>0.324</td>
<td>0.344</td>
<td>+6.2</td>
</tr>
</tbody>
</table>

**Table 5.2.** Mean Average Precision scores of the test set of pattern matching and SQR for QA pair corpus

<table>
<thead>
<tr>
<th>Training Set</th>
<th>Pattern Matching</th>
<th>SQR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Tf-idf</td>
</tr>
<tr>
<td></td>
<td>Baseline (k=4 without chunking)</td>
<td>K=3</td>
</tr>
<tr>
<td>QP(150)</td>
<td>0.458</td>
<td>0.374</td>
</tr>
<tr>
<td>QP(211)</td>
<td>0.461</td>
<td>0.379</td>
</tr>
<tr>
<td>QP and QA corpus</td>
<td>0.382</td>
<td>0.414</td>
</tr>
</tbody>
</table>

**Table 5.3.** Mean Average Precision scores of the test set of pattern matching and SQR for Search Engine
In our evaluation result, we are willing to investigate the effect of pattern matching and syntactic structure based query reformulation upon the retrieval performance from both question answer corpus and we have calculated MAP value either on the list of candidate answers or documents for QA corpus and Web respectively.

Table 5.2 shows the experimental result for both pattern matching and structural query reformulation on QA corpus. The size of the training set for structural query reformulation is same as the size of pattern corpus used for pattern matching. It is obvious from the table that pattern matching outperforms in reformulating query. For structural query reformulation, we have analyzed the effect of the size of POS block used to draw shallow syntactic evidence for retrieval process. It is evident from Table 5.2, that SQR remarkably improves retrieval performance compared to the baseline. In similar manner, we have evaluated the experimental result for both pattern matching and structural query reformulation for Web as the information resource in Table 5.3. It is evident from the table that pattern matching approach is giving much better result than structural query reformulation. However, in case of Web based searching, recall is lower for both pattern based and SQR method as natural language offers so many different ways to express answers. The table also indicates that when we vary the value of k i.e. number of tokens in POS blocks, the MAP value records low for the value 4. The slight decrement in MAP value has been recorded due to the fact that sometimes POS block may contain

![Figure 5.6. Relation between MAP values and no. of tokens (k) in POS block](image)

In our evaluation result, we are willing to investigate the effect of pattern matching and syntactic structure based query reformulation upon the retrieval performance from both question answer corpus and we have calculated MAP value either on the list of candidate answers or documents for QA corpus and Web respectively.

Table 5.2 shows the experimental result for both pattern matching and structural query reformulation on QA corpus. The size of the training set for structural query reformulation is same as the size of pattern corpus used for pattern matching. It is obvious from the table that pattern matching outperforms in reformulating query. For structural query reformulation, we have analyzed the effect of the size of POS block used to draw shallow syntactic evidence for retrieval process. It is evident from Table 5.2, that SQR remarkably improves retrieval performance compared to the baseline. In similar manner, we have evaluated the experimental result for both pattern matching and structural query reformulation for Web as the information resource in Table 5.3. It is evident from the table that pattern matching approach is giving much better result than structural query reformulation. However, in case of Web based searching, recall is lower for both pattern based and SQR method as natural language offers so many different ways to express answers. The table also indicates that when we vary the value of k i.e. number of tokens in POS blocks, the MAP value records low for the value 4. The slight decrement in MAP value has been recorded due to the fact that sometimes POS block may contain
inappropriate text segment(s) to render a complete query. It is also apparent from Figure 5.6 that both weighting models tf-idf and okapi BM25 are showing better MAP scores for information retrieval through Web as query reformulation using text structure is more viable to the Web rather than query answer pair corpus which relies on matching of somewhat predefined text structures.

5.6 Answer Retrieval from Web

We cannot always rely on corpus for answer retrieval as they are incorporated only to enhance the overall performance of QA system by providing almost static type of answers. For rest of the questions, we have to opt for retrieval through Web. Therefore, achieving a satisfactory performance in our question answering task requires integration of multiple strategies to deal with different characteristic of various information resources. Answer extraction from QA corpus has already been discussed in previous section. Now we will put our focus on Web-based extraction module of our proposed system.

The task of automated candidate answer extraction from Web requires pre-fetching of documents or snippets before actual answer extraction. After, pre-fetching of documents or snippets one often need to use the applications from deep or shallow NLP techniques, pre-stored knowledge information, language modeling and pattern mining to extract exact answer string. More recently, many of the question answering system either of open or restricted domain have used external information extraction such as surface text patterns for answer finding. We have also explored these surface patterns for our answer extraction phase with slight modification.

Our assumption is that, in text based question answering system, structural arrangements of POS tags can be used efficiently to identify answer string along with words and special user defined tokens. We have considered sequential patterns composed of POS (part of speech) tags which increases the likelihood of fetching more relevant information. We believe that shallow syntactic information in form of pattern will also improve the likelihood of fetching more relevant documents even after erroneous question classification.
QA system needs to analyze the natural language question presented by user, understand the question type and its focus word. Keywords and reformulated query have also been generated from previous phases which would help in answer extraction. Answer extraction from Web requires a reformulated query either from pattern matching or SQR based approach to be posed on search engine interface. Named Entity tagging has also been performed to keep track of semantics of the reformulated query and anticipated answer string. The query will then submit to the search engine (Google) interface to retrieve top 40 snippets. The snippets contain the URL, the titles, and some sting-segments of the related Web document. It will save much time to use snippets than download the whole pages. The experiment result indicates that the top 40 snippets are enough to gain the answer.

5.6.1 Existing Work

There has been a considerable amount of work on using Web information and search engines for TREC QA, owing to the fact that a data collection such as the TREC corpus has considerably less answer redundancy than the Web and thus, it is easier to match a question to the Web data, to extract answers from the matching text, and then project these answers on the restricted data collection (e.g. Brill et al., 2002; Radev et al., 1999; Ramakrishnan et al., 2004).

Most of the existing Web-based QA systems aim to English language and implement sophisticated natural language processing tools and/or large hierarchies of answer matching rules and answer types. Kwok et al., (2001) has investigated how to scale current paradigms to general QA on the Web. The system developed by Lin et al., (2002) also relied on Web to extract answer for factoid question. The benefit that the Web offer to the underlying architecture of QA systems have been earlier exploited by the existing systems (Clarke et al., 2001; Dumais et al.,2002)

Cucerzanet. al.,(2005) has given separate importance to structured information available on Web i.e., html tables on the Web to retrieve answers to factoid questions. By exploiting the explicit tabular structures created by the Web document authors, they acquired natural language understanding “for free” and hence, advance the applicability and scalability of question answering. There have been also other efforts to extract structured information from the Web. Previous approaches (e.g., Agichtein and
Gravano (2001), Etzioni et al. (2004)) focused on extracting specific relationships (e.g., "is a"), which can then be used to answer the specific questions that these relationships support (e.g., "who is X").

The huge amount of readily available unstructured data on Web provides data redundancy, which can be implemented with simple pattern matching techniques for candidate answer extraction from the fetched document list by search engine. Web search engines such as Google provide a convenient front-end for extraction and filtering of enormous amounts of Web data. Lin and Katz, (2003) have identified this class of techniques as the knowledge mining approach to question answering.

5.6.2 General Architecture for Answer Extraction

The proposed architecture can able to automatically give answer to the questions about higher education using natural language. The fundamental architecture of the answer retrieval stage is shown in Figure 5.8 and consists of question analysis, the Question Answer Corpus Retrieval, the Web retrieval processing and the preprocessing, and some external resources.

The user poses question to the system and in return get the corresponding answer. In the question analysis module, some processing is performed which involves morphological analysis, question classification, extraction of keywords and question reformulation. The answer retrieval module consists of the processing of question answer corpus retrieval and Web document retrieval and will be introduced later.

Pre processing of text is carried out before actual analysis, Tokenization, Pos tagging and chunking is performed before question classification. Question classification task provides question class and focus word at primary level and answer class at secondary level. After question classification, keywords are extracted and reformulated query is presented either with the help of pattern matching or structure based query reformulation as stated by Singh et al. (2016).

In the current system, the retrieval processing consists of three retrieval processes that include three retrieval processes which consist of similar question pattern retrieval, similar document retrieval and similar answer retrieval from the document.
While the Web is undoubtedly a useful resource for question answering, it is not without shortcomings. Owing to the large size of Web, useful information is often pushed behind by irrelevant materials and statistical techniques alone are not capable of refining right answer. To overcome these issues, there requires shallow NLP techniques, syntactic evidences and lexical disambiguation while QA processing. Here, in our proposed work, for answer extraction, we have presented a similarity measurement approach which determines the proximity between user’s question and answer based on blend of syntax, semantics and statistical schemes for answer candidate extraction in our higher education domain.

### 5.6.3 Similarity between Question and Answer

Calculating similarity of the question against answer is the basic criterion of retrieving answer candidates from Web. It directly affects the accuracy of answer extraction module. Meanwhile, many similarity measurements have been used by QA research community. Usually, it has been measured according to three aspects: syntax similarity, semantic similarity and pragmatic similarity. Pragmatic similarity is quite difficult and so far has not gained much satisfactory results. Semantic similarity has been calculated on the basis of same words (Nirenburg et al., 1993), based on relative dependency (Bing et al., 2003; Li et al., 2002), based on semantic dictionary (Bing et al., 2003) and based on editing distance etc. Among these methods similarity based on only word is not much effective without considering synonymous words. With the use of external semantic
resource like WordNet, the issue can be well addressed. But only using this method is not going to leverage required accuracy as it doesn’t consider the structural relationship between the entities present in text.

Based on the existing issues of the similarity calculation, we have combined the word-based similarity with syntax and semantic-based similarity together along with rank of the document returned by the search engine i.e., Google. The search engine returns a set of ordered list of documents along with snippets which are supposed to contain an answer to the posed query by referring to those documents. We have gained the top 40 extracted document list through search engine retrieval, and then process the snippets in the results and form a candidate snippet set according to OkapiBM25 similarity metric as mentioned in previous section of query reformulation. Secondly, POS tag and chunk the sentences of the snippets. The retrieved snippets are further analyzed to extract answer to the original question on the basis of specifications generated by the question analysis component such as question class, focus word etc. During analysis, number of candidate answers are generated which are further sent to answer extraction component. The candidate snippet set requires to be analyzed through different similarity measures given below to extract the exact answer string:

- **Word order similarity**

  The user question transformed to query is initially considered as set of words and these words along with their order of occurrence are used to match with the candidate snippets retrieved from search engine. Word order similarity between question and answer has been calculated by the formula proposed by Li et al. (Li et al., 2006) for measuring word order similarity as given by Equation 5.2:

  \[ R_i(Q,D) = \frac{1}{r_i} \]

  However, it is not necessarily true that all snippets returned by the search engine are relevant and contains the desired answer but simultaneously, it is also true that modern search engines apply sophisticated techniques and page ranking algorithms to determine rank of the particular document hence snippets. Therefore we can use
document rank as reliable measure as indicator of characterizing nature of the snippet.

- **Similar text pattern count across snippets.**
  Owing to the POS tag based reformulated query, the QA system knows what the expected answer pattern likely to be. A count is added to calculate same answer segment across the retrieved snippets for the query to find most frequent answer segment, i.e., $C_i$. The score associated with each answer is boosted up based on the number of results returned by the search engine for each queries.

- **Semantic similarity**
  This is a feature that indicates whether a word and its synonym appear in an answer candidate and a question respectively. We use WordNet to determine semantic similarity by the Wu & Palmer’s Measure (1994) given as:

$$S_i(Q,A) = \frac{2 \times \text{depth}(\text{Iso}(Q,A))}{\text{len}(Q,A) + 2 \times \text{depth}(\text{Iso}(Q,A))} \tag{5.4}$$

Where $\text{len}(Q,A)$: the length of the shortest path from synset $Q$ to synset $A$ in WordNet.
 Iso $(Q, A)$: the lowest common subsumer of $Q$ and $A$
 depth $(c_i)$: the length of the path to synset $c_i$ from the global root entity, and $\text{depth}$(root)=1.

Then the total similarity between question and answer is:

$$\text{Sim}_i(Q, A) = \lambda_1 W_i(Q, A) + \lambda_2 R_i(Q, A) + \lambda_3 C_i(Q, A) + \lambda_4 S_i(Q, A) \tag{5.5}$$

Where $\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1$. Apparently $0 \leq \text{Sim} (A, Q) \leq 1$. The similarity between question and answer is decided together by the above factors, the influence of $W_i(Q, A)$ is dominating, and the other factors have accessoril function. According to the testing results to some questions, we define several rudimental parameters: $\lambda_1 = 0.6$, $\lambda_2 = 0.2$, $\lambda_3 = 0.1$, $\lambda_4 = 0.1$. 

97
After the question type has been identified, the system extracts all such type information from the snippets as candidate answers, using our based named entity recognizer as well as anticipated answer pattern. Then the system selects final answer with highest score. However, we have avoided a new search for a pattern having its synonym as every new pattern adds a substantial computation effort to the system. The representation of the question and the representation of the candidate answer-bearing texts are matched against each other and a set of candidate answers is produced, ranked according to likelihood of correctness based on above measures.

The process of answer extraction for our proposed system utilizing different information resources can be combined and presented in the form of following algorithm:

**Algorithm: Answer retrieval from QA corpus and Web**

**Input:** Question q

**Output:** Candidate answer list $C_a = \{a_1, \ldots, a_n\}$ / documents $C_d = \{d_1, \ldots, d_n\}$

**Begin**

1. Apply a POS tagger and chunker to q.
2. Classify question q.
3. Reformulate q to query $q_r$.
4. Match answer class $C_a$, coarse grained class $C_{qc}$ and fine grained class $F_{qc}$ and focus word $F_{qc}$ to refine candidate queries $q_i$ from QA corpus.
5. Calculate matching similarity $S_i$ using equation (1).
6. If $S_i \geq \lambda$, i.e., threshold value, select answer of top k queries as answer candidates.
7. Else, forward query to search engine interface and extract top n snippets.
8. Calculate matching coefficient $W_i(s_i, q_r)$, $Sem_i(s_i, q_r)$, $R_i(s_i, q_r)$ and $C_i(s_i, q_r)$ to calculate total similarity.
9. Estimate total similarity $S_i(s_i, q_r)$ using equation (4).
10. Fetch document list.

**End**

Figure 5.8. Answer Extraction Algorithm
5.6.4 Experiment and Evaluation

The performance of the mentioned similarity based answer extraction method has been evaluated using the 364 question of higher education domain collected from different university Websites, online educational forums and communities etc. These questions cover each type in the proposed taxonomy. The number of questions belonging to each class is shown in Table 5.4.

In our work, we consider MRR as performance evaluation criterion. The mean reciprocal rank is a statistic measure for evaluating the possible response list produced by the QA system against sample of questions ordered by probability of correctness. The reciprocal rank of a question response is the multiplicative inverse of the rank of the first correct answer. The mean reciprocal rank is the average of the reciprocal ranks of results for a sample of questions is given by

\[
MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i}
\]  

(5.6)

where, \(rank_i\) refers to the rank position of the first relevant answer/document for the \(i^{th}\) question.

<table>
<thead>
<tr>
<th>Coarse class</th>
<th># of Question</th>
<th>MRR</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Web</td>
<td>Corpus</td>
</tr>
<tr>
<td>Abbr</td>
<td>17</td>
<td>0.345</td>
<td>0.389</td>
</tr>
<tr>
<td>Entity</td>
<td>39</td>
<td>0.334</td>
<td>0.364</td>
</tr>
<tr>
<td>Loc</td>
<td>48</td>
<td>0.328</td>
<td>0.375</td>
</tr>
<tr>
<td>Person</td>
<td>50</td>
<td>0.381</td>
<td>0.395</td>
</tr>
<tr>
<td>Number</td>
<td>36</td>
<td>0.332</td>
<td>0.345</td>
</tr>
<tr>
<td>Org</td>
<td>38</td>
<td>0.345</td>
<td>0.39</td>
</tr>
<tr>
<td>Desig.</td>
<td>10</td>
<td>0.297</td>
<td>0.365</td>
</tr>
<tr>
<td>Description</td>
<td>63</td>
<td>0.305</td>
<td>0.333</td>
</tr>
<tr>
<td>Time</td>
<td>65</td>
<td>0.355</td>
<td>0.37</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>364</strong></td>
<td><strong>0.336</strong></td>
<td><strong>0.370</strong></td>
</tr>
</tbody>
</table>

Table 5.4. MRR score for Answer Extraction from QA corpus and Web
Table 5.4 shows the results achieved by answer extraction mechanism. As, it can be fairly observed from Figure 5.10 that extraction from QA corpus shows better MRR score than obtained from Web. However, for some classes like person, number and time our integrated approach achieves equal MRR score for both the resources. Overall extraction performance from Web slightly lags in MRR score but it is fairly acceptable owing to large size and presence of noise information at Web. The overall MRR score achieved by our extraction approach in higher education domain is 0.336 and 0.370 for Web and question answer corpus respectively which indicates its satisfactory performance. Our belief to integrate Web-based extraction to Corpus based has sustained as both the strategies played a complementary roles in enhancing the performance.

### 5.7. Summary

In proposed chapter, we have implemented query reformulation ideas with POS tag patterns. SQR is also implemented using syntactic blocks composed of POS tokens as secondary reformulation approach. Our experiment shows encouraging result for the reformulation task. Thereafter, we have presented answer extraction task based on similarity measurement calculation and achieved satisfactory MRR for Higher Education Domain.