Bengali Monolingual Question Answering System

A Dissertation

Submitted in partial fulfillment of the requirements
for the degree of

Máster Universitario en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

by

Somnath Banerjee

Advisors:

Prof. Paolo Rosso
Prof. Sivaji Bandyopadhyay
Dr. Sudip Kumar Naskar

UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

Departamento de Sistemas Informáticos y Computación
Universitat Politècnica de València

July 2018

© July 2018, by Somnath Banerjee
All rights reserved

To my ***Parents...***

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I declare that I have properly and accurately acknowledged all sources used in the production of this thesis.
I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Somnath Banerjee

Date: July 15, 2018

Acknowledgements

The completion of this thesis would not have been possible without the support and encouragement of my advisor, colleagues, family and friends.
In particular I want to thank my advisor, Prof. Paolo Rosso. In spite of his tight schedule, he gave me his time every week, advised me, helped me, encouraged me and never blamed me a single time for my wrongdoing.
Special thanks go to Prof. Sivaji Bandyopadhyay, who first introduced me to Natural Language Processing back in India and kindled my research interest in the field.
I would like to thank Dr. Sudip Naskar for his guidance. He showed me the importance of good writing in research.
I am forever indebted to Prof. Manuel Montes-y-Gómez for his challenging questions that allowed me to focus on the fundamentals and Dr. Francisco Rangel for sharing his knowledge.
On a more personal level, I would like to thank Simona, Bilal, Maria, Andrés, and Juan, as I have a great time in Valencia.
I would also like to thank Elena Taulet Grech for providing me the administrative support.
Last but certainly not least, I would like to acknowledge the financial support that I received from the Erasmas Plus KA 107.

Somnath Banerjee
Universitat Politècnica de València
July 15, 2018

Abstract

Since the middle of the 20th century, question answering (QA) serves as one of the primary research areas in natural language processing. QA is the task of automatically generating answers to natural language questions from humans. This dissertation focuses on answering to the Bengali factoid questions.

Bengali (also known as 'Bangla'), the seventh most spoken language in the world, is mainly spoken in the eastern states of India, namely, West Bengal, Assam, Tripura, and Andaman and Nicobar Islands. Bengali is recognized as one of the official languages of India and it has approximately 208 million native speakers. Also, Bengali is the official language of Bangladesh.

Prior to this research work, only a few research works have been carried out in other Indian languages. Even there did not exist any QA system for Bengali. Moreover, any dataset was not available for QA research in Bengali.

In this dissertation, we present an approach to develop a Bengali QA system which deals with factoid questions. Although the proposed methodologies of this QA research has been carried out in Bengali, the approach could be used for developing QA system in other Indian languages as well.

In this work, we propose an annotation scheme for Bengali QA dataset. The dataset contains 47 documents along with 2,257 questions from three domains, namely history, geography and agriculture. Particularly for Bengali QA, we develop a named entity identification system which is based on Margin Infused Relaxed Algorithm (MIRA). The system obtains 91.23%, 87.29% and 89.69% precision, recall and F-measure, respectively.

We propose a question taxonomy that consists of 9 coarse-grained along with 62 fine-grained question classes for Bengali. For classifying Bengali questions, we propose two machine learning approaches: individual classifier based approach, and a classifier combination approach. Finally, we propose a framework for answering factoid Bengali questions, namely *BFQA*.

Contents

List of Figures

List of Tables

Introduction

The World Wide Web (WWW) has grown dramatically since its inception in 1992 as a global interconnected system (Berners-Lee et al., 2010). Internet users can now find relevant information through information retrieval systems (search engines, such as Google, Bing, etc.) by submitting queries. For instance, if we want to retrieve the birthday of Narendra Modi, then we have to search for the documents related to Narendra Modi and hope that the returned documents will contain the date of birth information of Narendra Modi. Certainly, it would be a much more convincing solution if one can submit the question '*What is the birthday of Narendra Modi?*' to the system and obtain the answer '*17th September 1950*'. Gradually, researchers understood the need of a system which can take queries in more specific as well as in natural language and instead of relevant documents it returns the exact answer or phrase that contains the answer. Thus, the system allows the user to pose natural language question as query and in response it returns a phrase or a short passage or even the exact answer. Such systems are referred to as Question Answering (QA) systems. The virtue of QA systems is that they save invaluable time of user's effort to satisfy an information need. Over time, several QA systems have been developed for a variety of purposes on different languages (mainly on English).

India (officially the *Republic of India*), an emerging superpower of the world, located in South Asia, is regarded as the second-most populous country of the world having more than 1.3 billion speakers with diverse languages. The constitution of India has identified 22 languages as the official languages of the country. Apart from the 22 languages, there are other languages too that are recognized by the Indian Government but not as the official languages. The officially recognized languages of this country further include various dialects and variations. Bengali, an Indian language mainly spoken in the eastern states of India, namely, West Bengal, Assam,

Tripura, and Andaman and Nicobar Islands, is the seventh most spoken language in the world. Bengali is recognized as one of the official languages of India and it has approximately 208 million native speakers. Also, Bengali is the official language of Bangladesh. Due to the recent digitization of the Indic languages, popular newspapers, news channels, government sites publish their contents in Indian languages including Bengali. Considering the superiority of QA systems over the Information Retrieval systems (i.e., search engines), there is a ever-growing need for QA systems in Bengali. However, till date no such QA system has been developed in Bengali. Such circumstances motivated us to concentrate on Bengali QA research.

1.1 Motivations and Objectives

Although QA research began its journey since the design of BASEBALL (Green Jr et al., 1961b) system, new QA systems are being developed continuously. The researchers still find a place of contributions to QA research. This is the one of the fundamental motivations of this thesis. In this regard, the high-profile campaigns are Text REtrieval Conference[1] (TREC), Conference and Labs of the Evaluation Forum[2] (CLEF). TREC is being organized since 1999. The major focus of TREC is on short, fact-based questions. On the other hand, CLEF is an European QA campaign similar to TREC and it mainly focuses on major European languages. One of the notable tasks in CLEF[3] is bilingual QA, i.e., query is in one language and the answer has to be retrieved from documents in another language.
Although the QA systems developed for European languages, particularly in English, have achieved reasonable accuracy, the situation for the Indian languages is completely different. Research on QA has not been initiated for most of the Indian languages. Like other Indian languages, Bengali (also known as 'Bangla') presents serious challenges for QA. Bengali is an Indo-Aryan language like Hindi, Marathi, Gujrati, etc. With about 189 million native speakers and about 208 million total speakers, Bengali is one of the most spoken languages (ranked seven) in the world and the second most commonly spoken language in India. However, any QA system is not available for Bengali. Therefore, one objective of this thesis was to create a QA system for the Bengali native speakers.
The research presented in this thesis has the following objectives:

- To conduct a detailed literature survey to understand the state of the art in the field of QA

[1] http://trec.nist.gov/
[2] http://clef2017.clef-initiative.eu/
[3] https://www.ercim.eu/publication/Ercim_News/enw55/clef2003.html

- To create language resources for monolingual
- To develop a factoid QA system for answering monolingual Bengali questions

1.2 Contributions

By achieving the objectives mentioned in the previous section, this research makes several contributions to the field of question answering. The contributions this thesis presents are listed below.

Bengali monolingual QA research:
Prior to this research work, there did not exist any QA system for Bengali. Even in other Indian languages, only a few research works have been carried out. Moreover, any dataset was not available for QA research in Bengali. Although the proposed methodologies of this QA research has been carried out in Bengali, the approach could be used for developing QA system in other Indian languages as well. While developing QA system on monolingual Bengali, the following contributions are made:

- A dataset along with an annotation scheme is proposed for Bengali QA research. This dataset is not only regarded as the first corpus for QA research in Bengali, it is the sole corpus till date. The dataset contains 47 documents along with 2,257 questions from three domains, namely history, geography and agriculture.
- A named entity identification system is developed which is based on Margin Infused Relaxed Algorithm (MIRA) particularly for Bengali QA. The proposed system obtains 91.23%, 87.29% and 89.69% precision, recall and F-measure, respectively.
- As part of the question analysis, a question taxonomy has been proposed which consists of 9 coarse-grained along with 62 fine-grained question classes for Bengali. Till date this is the only available taxonomy for Bengali QA research. For classifying Bengali questions, we have proposed two machine learning approaches: i) individual classifier based approach, and ii) a classifier combination approach.
- We introduce a framework for answering factoid Bengali questions, namely *BFQA*. Till now this is the only available factoid QA system in Bengali. Within the BFQA framework, sentence selection and ranking scheme are also proposed.

1.3 Research Questions

The aforementioned objectives can be divided into two groups according to the scenario where the QA system is to be employed: monolingual and code-mixed cross-script. With respect to these groups, we list the following research questions (RQ) that are investigated in this thesis.

Questions about the monolingual Bengali QA system

RQ-1: How to build the corpus for Bengali QA ? [Chapter 3]

RQ-2: How question analysis is useful with respect to answer retrieval strategy in monolingual Bengali QA? [Chapter 4]

RQ-3: What named entity retrieval strategy should be followed to identify the named entities for factoid questions posed in Bengali? [Chapter 5]

RQ-4: Are there any differences in the process of answer extraction between a Bengali QA systems and the other existing QA systems? [Chapter 6]

1.4 Structure of the Thesis

This thesis comprises of seven chapters in total. (i) We first introduce the research overview in Chapter 1. (ii) We survey the background literature for the thesis in Chapter 2. (iii) We discuss the procedure of corpora preparation in Chapter 3. (iv) As an integral part of QA, question analysis is discussed in Chapter 4. (v) In Chapter 5, we propose a model which identifies the named entities from Bengali QA corpus. (vi) In Chapter 6 we present a QA framework for Bengali. (vii) Finally, Chapter 7, summarizes the research presented in this thesis and discusses about the future directions that could be pursued along this line of work.

Background Review

We, the human beings are familiar with question and answer since our childhood. Because we learn by asking questions on the things that we encounter for the first time. However, we often think about how to express the question or how to properly answer the question that is asked. If we consult with the New Oxford Dictionary of English (2001), those are defined as below:
The *question* is a sentence worded or expressed so as to elicit information and its parts-of-speech is noun. The *answer* is noun as well and it is a thing that is said, written, or done to deal with or as a reaction to a question, statement, or situation. In this thesis, we start with the basic of question and answering and then we move to the earlier work of question answering.

Question: Technically the question can be expressed as 'a request for information'. Although in the written form of most of the languages we insert a question mark ('?') at the end of sentence, the questions are posed similarly with normal dialouge in the spoken form. However, there are languages (like Spanish) where a question is formed by putting two question marks: one at the beginning and another at the end. Therefore, questions are recognised easily in the written form than the spoken act/form. However, the request for information (i.e., question) can be expressed without using a question mark. For example, the question "*Which cities have metros in India?*" can be asked as "*Name the cities in India which have metros.*" Although a question can be asked in the form of instruction, we treat it as a question.

Answer: For the sake of simplicity, we can say that the answer is 'a response to the request for information'. Because it is given in response to a question. However, the answer may be correct or not. It is not straightforward to judge the correctness

of an answer. Many attempts were taken to define a correct answer to a question. Voorhees et al. (1999) stated that an answer is defined as a string of up to 50 or 250 characters in length which contained a correct answer in the context provided by the document; Voorhees (2003) quoted that an answer is judged correct if the retrieved document contains the correct answer; As per Roberts and Gaizauskas (2004), an answer is a text snippet which can be extracted from the relevant document.

Question Answering: In this present digital era, people are searching for their every information need from the ever growing digital documents using the well-known information retrieval (IR) system. The most widely used IR system is the online search engine (e.g., Google). In the traditional IR systems, we usually type a few keywords and the IR system returns a list of documents based on the relevancy to the user's information need. Then, the user has to scan the documents for pertinent information. Although posing and retrieving the relevant documents are easy from the user's perspective, reading each document throughly to extract the answer is not only time consuming but also it takes an exam of the user's patience if the answer is not find in the top ranked documents (Moldovan and Surdeanu, 2002). Finding the exact answer to the user's query bring into the world the emergence of the Question Answering (QA) research. The task of QA is a subtask of Natural Language Processing (NLP) which is concerned with answering the questions posed in a natural language. According to Maybury (2004):

"*Question answering is an interactive human computer process that encompasses understanding a user's information need, typically expressed in a natural language query; retrieving relevant documents, data, or knowledge from selected sources; extracting, qualifying and prioritizing available answers from these sources; and presenting and explaining responses in an effective manner.*"

The difference between the traditional IR and QA is shown in Table 2.1.

Table 2.1: QA vs IR

System	Input	Output
IR	Keywords	Ranked Document list
QA	Question in Natural Language	Exact answer with/without supporting text

This chapter presents previous and recent methods relevant to this research project. The QA research has been started since 1960 and a number of systems have been developed till date (Androutsopoulos et al., 1995). The present systems concentrate on answering the questions from different data source such as sementic web (Dwivedi and Singh, 2013; Kumar and Zayaraz, 2015; Lopez et al., 2011), social media data, etc. Even the format of answers is being shifting from simple text to

multimedia(Burger et al., 2001). Two of the best-known early QA systems were BASEBALL (Green Jr et al., 1961a) and LUNAR (Woods, 1973). The BASEBALL system was designed to answer questions about baseball games which were played in the American league during a single season, Green Jr et al. (1961a) proposed BASEBALL that answers to the questions about the American baseball league during a particular season. BASEBALL could answer to the questions related to dates, location, etc. Woods (1973) proposed LUNAR that provides information about the Apollo moon mission. Both the systems transform users' questions into database queries through plain pattern matching rules and finally generates the answers. This pattern matching approach strongly depends upon the application domains. Considering the diversity of a natural language in the form of paraphrasing, pattern matching approach was not a feasible solution. Although the systems like BASEBALL, LUNAR, etc. performed well, the systems were limited to the particular domain with access to a structured database containing the available domain knowledge. Typically the questions were transformed into standard database query to access the database repository. The QA systems developed during the 1970's followed the approach employed by BASEBALL and LUNAR (Grosz et al., 1986).

2.1 Question Answering Systems

Since its inception from 1960, many approaches were employed to develop QA. Not surprisingly, since the last few years, deep learning approaches are trying to fit into QA research. Survey papers for QA are recurrently authored because of its ample publications and rapid progress. Recently, Mishra and Jain (2016) classified the QA approaches based on eight parameters: (1) application domains for which QA systems are developed, (2) types of questions asked by the users, (3) types of analyses performed on users' questions and source documents, (4) types of data consulted in data sources, (5) characteristics of data sources, (6) types of representations used for questions and their matching functions, (7) types of techniques used for retrieving answers (8) forms of answers generated by QA systems. Whereas, few years back, Dwivedi and Singh (2013) quoted that QA systems follow three approaches: linguistic approach, statistical approach and pattern matching approach.
The following is a few of them on different areas of question answering:

2.1.1 Open-domain vs Restricted-domain

Based on the target domain of interest, the QA systems can be broadly classified into two: open-domain QA and restricted-domain QA. However, Harabagiu and Moldovan (2003) argued about three: canned, closed-domain and open-domain.
Canned QA Systems are the simplest type of QA systems because they do not answer questions automatically, but instead rely on a very large repository of ques-

tions for which the answer is known. Extending coverage with new questions and their answers relies on human effort. To answer unseen questions, systems usually retrieve the answer of the most similar existing question. They can be useful in restricted domains, where users' information needs are predetermined, e.g., a help desk.

This type of system originated from lists of Frequently Asked Questions and bulletin board systems, such as FidoNet and UseNet. Web 2.0 technologies enabled the development of social QA Forums, such as Yahoo!Answers[1], WikiAnswers[2], Stack Exchange[3] and Quora[4]. In these collaborative systems, users post new questions and contribute answers to existing ones, thus broadening the coverage of the systems. To increase recall and eliminate inconsistencies caused by duplicates, each question may have an associated set of alternative phrasings. Users also rate the correctness and utility of the answers. These repositories are usually available on the Web and users can simply employ their favourite Web search engine to find out if their question already has an answer. Their advantage is that most questions and their answers cover complex information needs which require human-like cognitive skills.

The main contribution such systems make to QA research is the amount of data they provide, which could be used in developing and testing the automatic QA systems. They can also be exploited to investigate question similarity metrics to establish when different questions ask the same thing. A good similarity metric can distinguish whether two questions are equivalent or not, for example when the word order differs or when synonyms or rephrasing are used, or whether the questions ask for different information.

Open-domain QA focuses on answering questions regardless of the subject domain. Extracting answers from a large textual corpus of textual documents is a typical example of an ODQA system (Maybury, 2004). Therefore, theoretically these systems can answer any topic using any textual collection. Considering the fact that information availability is rapidly growing on the Web, these QA systems could act as an alternative to the search engines. Despite of the substantial research that carried out in the last decade, open-domain QA systems have not become an alternative to Web search engines. The first Web-based QA system was *SynTactic Analysis using Reversible Transformations* (START[5]) by Katz (1997). QA systems based on open-domain, make use of general ontology and world knowledge to answers (Lo and Lam, 2006). Indurkhya and Damerau (2010) criticized that usually casual users use

[1] https://answers.yahoo.com/
[2] https://www.wikianswers.com/
[3] https://stackexchange.com/
[4] https://www.quora.com/
[5] http://start.csail.mit.edu/

open-domain QA systems and the quality of answers is not well. Example of open-domain QA systems are Webclopedia (Hovy et al., 2000b), Mulder (Kwok et al., 2001), Answerbus (Zheng, 2002b), etc.

In contrast to the open-domain QA systems, **Restricted-domain or closed-domain QA** systems are developed to answer questions posed for a specific domain and usually answers are searched within domain specific document collections (Mollá and Vicedo, 2007). These systems make use of domain specific ontology and terminology and generally the question patterns are very limited. The advantage of this type of system is that it deals with very specific data which usually does not contain ambiguous terms and as a result can be processed more easily. Hence, the systems can achieve satisfactory accuracy and the quality of the results is high as well. There are various restricted domain QASs developed so far such as: geospatial domain, medical domain, community based, etc. Indurkhya and Damerau (2010); Lopez et al. (2011) reported that the different restricted domain QA systems can be combined to build general domain QA systems. However, the systems should have the capability to assign the question to the relevant QA system based on the knowledge derived from the keywords of the question. Examples of restricted-domain QA systems are BASEBALL, WEBCOOP (Benamara, 2004), etc.

2.1.2 Question Types

Li and Roth (2002a); Moldovan et al. (2003) argued that the performance of the QA systems are highly dependent on the type of the question asked by the users' and results confirmed that 36.4% of errors occur if the questions are classified wrongly. A question may be classified into one of the different categories: factoid (e.g., "What is the capital of Spain?", list (e.g., "List the countries that have won the World Cup."), causal (e.g., "What causes teenagers to spend time in Facebook?", yes-no (i.e., "Did you go to the concert?"), definition (e.g., "What is DNA?") , procedural (e.g., "How to prepare tea?"), etc. Each question type needs specific strategy to handle.

Factoid QA is the most widely studied task in QA. Typically, factoid questions are simple and fact based that require answers in a single short phrase (Indurkhya and Damerau, 2010). Strzalkowski and Harabagiu (2006) reported that present QA systems can now answer over 70% of arbitrary, open domain factoid questions. Over time, to answer the factoid questions, a number of approaches were adopted. Ravichandran and Hovy (2002) proposed a rule based approach to answer factoid questions by learning the surface patterns of the questions. The survey of Kolomiyets and Moens (2011); Lopez et al. (2011) found that the expected answer types for the factoid type questions are generally named entities which could be traced in corpus through named entity tagging. Grappy and Grau (2010) argued that the Wikipedia or news wire text can be used as a corpus repository for the factoid QA systems.

List questions asks for different instances of a particular type, i.e., the set of all such distinct instances in the corpus. Therefore, the answer to a list question consists of an unordered set of instances. Similar to factoid questions, the expected answer types are named entities for the list type questions. Hence, the successful approaches to factoid questions can work well for dealing with list type questions. A few examples of list QA systems are: Start (Katz et al., 2002), WEBCOOP (Benamara, 2004)

Causal QA is also known an 'Why QA'. In causal QA, the task of the QA system is to retrieve answers from a given text archive for a why-question. The answers are usually text fragments consisting of one or more sentences. Oh et al. (2013) reported that the performance of remains much lower than that of the state-of-the-art factoid QA systems, such as IBM's Watson (Ferrucci et al., 2010).

Yes-no QA is formally known as a polar question. The expected answer of this type of question is either 'yes' or 'no'.

2.1.3 Language Paradigm

Since NL questions and answers are represented by language, we can characterize the QA systems by the source language that represents questions and the target language that represents answers.

Monolingual QA system: In Monolingual QA systems, both questions and answers are in the same language. Monolingual QA is good for those people who speak one of the popular languages and researchers have paid a great deal of attention to monolingual QA research. Monolingual system is built to rely on as few resources as possible.

Cross lingual /Translingual QA system: In Cross lingual QA (CLQA), the question is posed in a source language and the answer must be found in a target collection of a different language. Most attempts at CLQA have so far concentrated on translating the query into English and performing monolingual English QA on the translated query.

QRISTAL (Laurent et al., 2005)(French acronym for "Question Answering Integrating Natural Language Processing Techniques") is a cross lingual question answering system for French, English, Italian, Portuguese, Polish and Czech. It was designed to extract answers both from documents stored on a hard disk and from Web pages by using traditional search engines (Google, MSN, AOL, etc.).

Multilingual QA System: In Multilingual QA (MLQA) system, user asks questions in one language and gets answers which are different from the source language

or same as the source language. To design such a system along with the technical details, there is a need to concentrate on the linguistic perspective. Multilingual QA has emerged only in the last few years as a complementary research task, representing a promising direction for at least two reasons. First, it allows the users to interact with machines in their native languages, contributing to easier, faster, and more equal information access. Second, cross lingual capabilities enable QA systems to access information stored only in language-specific text collections.

2.2 A Typical QA System Architecture

Harabagiu and Moldovan (2003) reported that QA is not just a document retrieval as it provides a small set of exact answers and a QA system made up of three modules: a question processing module which understands what the question is asking, a document processing module which finds relevant textual source, and an answer extraction and formulation module which extract the exact answer from the textual source. Hirschman and Gaizauskas (2001) also provided a comprehensive description of the QA architecture. However, the architecture of a QA system may be complex based on the context/requirement such as cross-lingual QA (cf. 2.1.3) or multilingual QA (cf. 2.1.3). A general architecture is given in Figure 2.1.

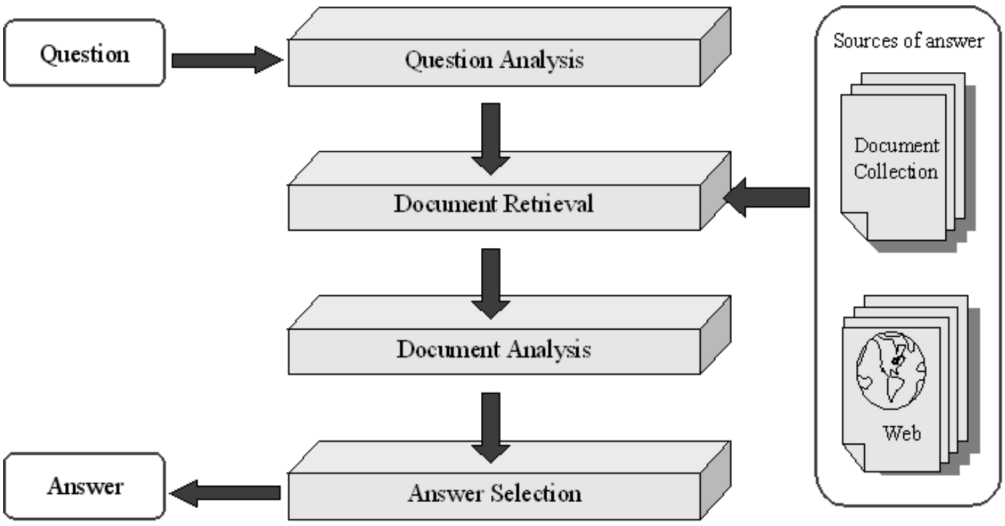

Figure 2.1: QA architecture (Hu, 2006)

It is observed from the QA pipeline (c.f. Figure 2.1) that the natural language questions feed into the first module (i.e., question analysis module) and after some steps the user gets the answers from the last module.

The first module is the **question analysis** that is concerned with processing input questions which is asked in a natural language (e.g., Bengali) by a user and determining the question type or question class, the expected answer type, the question

focus and what other entities and/or keywords are present. Question type is determined by using the question taxonomy. In question analysis, expected answer type (EAT) is also determined. While processing the textual source, the EAT is used for evaluating the textual source. Only the textual source (e.g., a sentence) containing an entity of the expected type will be processed further. Usually, the question focus indicates which entity the question is about, e.g., '*Narendra Modi*' in '*When was Narendra Modi born?*'. The question analysis information (such as question class, EAT, etc.) are used to make a intermediate query which is passed to the next modules.

Document retrieval module takes the query as input that is built in the previous stage. Usually, the retrieval module identifies a subset of documents that contain terms of a given query from the total document collection. This module can make use of different types of linguistic annotations. The documents can be annotated during indexing with lexical information (words, stems, lemmas), syntactic information (chunks, multi-word expressions, dependency trees), named entities annotations and semantic role labels or discourse structures. These informations can be used at the retrieval stage for improving the relevance of the retrieved data. However, if a large or dynamic collections (such as the Web) is used, it is impractical to perform these annotations on all documents. Therefore, such annotations are performed after retrieving the useful documents. This module returns a set or ranked list of documents that most likely contains the answer to the user's question. And then they are further analyzed by the document analysis component.

The **document analysis** module takes as input the documents that are suggested by the previous module. Also, it receives the useful information for selecting the word(s) or phrases that may count as candidate answers. This useful information is generated during the question analysis. This module extracts a number of candidate answers which are passed to the next module.

The **answer selection** module selects the most likely candidate answers which are extracted in the previous modules. The candidate answers are evaluated using the information gathered in the question analysis module. The candidate answers are ranked according to the different measures of similarity to the question (Harabagiu et al., 2003; Moldovan et al., 2007). One of the most common approaches to identify the correct candidate answer is pattern-based approach. Generally, these patterns are learned from the training data (Schlaefer, 2012; Schlaefer et al., 2006). Answer filtering is another approach which uses EAT to filter out the incompatible candidate answers. Often, answer validation approaches are used to determine the correctness of the extracted answer. (Moldovan et al., 2007) used textual entailment and logic formalisms to validate answers. Another alternative can be the edit distance between the question and the answer as a measure of similarity.

2.3 QA Evaluation

Evaluating an NLP system is a highly subjective task. Like most of the NLP task, there is not a single correct answer. Therefore, it is necessary to define what constitutes an answer to a question. Comprehensibly the users accept an answer when it gives relevant information. However, it still leaves a lot of scope for different systems to present the same answer in different ways. Hence, the evaluation of QA systems depends on the criteria for judging an answer. For evaluating an answer, the followings are some potential criteria (Hirschman and Gaizauskas, 2001):

1. Relevance: The answer should be a response to the question.
2. Correctness: The answer should be factually correct.
3. Conciseness: The answer should not contain extraneous or irrelevant information.
4. Completeness: The answer should be complete, i.e., not a part of the answer.
5. Justification: The answer should be supplied with sufficient context to allow a user to determine why this was chosen as an answer to the question.

While an answer extracted from the document repository, the answer will get one of the three distinct judgments based on the previously mentioned criterion.

- "Correct": The answer satisfies the criterion 1 & 2, i.e., the answer is responsive to a question in a correct way.
- "Inexact": The answer satisfies the criterion 3 & 4, some data are missing from or added to the answer
- "Unsupported": The documents do not support the extracted answer (i.e.,criteria 5).

TREC QA Track (Voorhees, 2002), defined two types of answers: chunks of 50-bytes (called '*short answers*') and chunks of 250-bytes (called '*long answers*'). For each question, the automatic evaluation requires the pair ["correct answers patterns", "supporting documents identifier"]. The evaluation is said to be '*Lenient*' if it uses only the answers patterns without using the supporting documents identifiers, and hence it does not ensure that the document has stated the answer. In contrast, the evaluation is said to be '*Strict*' if it uses both the answers patterns along with the supporting documents identifiers.

Although different QA evaluation campaigns (e.g. TREC, CLEF, NTCIR, etc) proposed several performance metrics, the most commonly used measures for automated evaluation are as follows:

Precision and ***recall*** and ***F-measure*** metrics are widely used in IR. Precision is the measure of accuracy. Recall is the measure of exhaustivity. The F-measure is the weighted harmonic mean of the precision and recall.

Precision indicates how many of the answers are correct. It is computed as:

$$\text{precision} = \frac{\text{number of correct answers}}{\text{number of questions answered}}$$

For each question, there is an expected set of correct answers, and these are called the gold standard answers. Recall indicates how many of the returned answers are in the gold standard. It is computed as:

$$\text{recall} = \frac{\text{number of correct answers}}{\text{number of questions to be answered}}$$

We explain these parameters using the following example. Assume a user asks the question: "Which are the three primary colors?". The gold standard answers would be "green", "red" and "blue". If a system returns "green" and "blue", as the answers, then the precision is 1, since all answers are correct, but the recall is only 2/3 since the answer "red" is missing.

$$\text{F-measure} = \frac{2(\text{precision} \times \text{recall})}{\text{precision} + \text{recall}}$$

The ***Mean Reciprocal Rank*** (MRR) was first used for TREC-8 (Voorhees et al., 1999) and it is used to evaluate the rank of the answer (relevance). MRR provides a method for scoring systems which return multiple competing answers per question. Let Q be the question collection and r_i the rank of the first correct answer to question i or 0 if no correct answer is returned. MRR is then given by:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{r_i}$$

Voorhees et al. (1999) reported that MRR has a number of drawbacks: (i) no credit is given to the QA systems for retrieving multiple (different) correct answers; and (ii) no credit is given to the QA systems for determining that it do not know or can not locate an appropriate answer to a question.

Later, it was realised in TREC that instead of evaluating the QA systems over multiple answers per question the evaluation should be based on a single exact answer per question. Hence, a new evaluation metric ***confidence weighted score***

(CWS) was introduced in TREC-11 (Voorhees, 2002). Under this evaluation metric, a QA system returns a single answer for each question. Therefore, before evaluation, the extracted answers are sorted so that the best answer (system has most confidence in) is placed first and the last answer be the one that the system has least confidence in. CWS is formally defined as:

$$CWS = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{\text{number correct in first i answers}}{i}$$

Thus, CWS rewards systems that can not only provide correct exact answers to questions but also that can recognise how likely an answer is to be correct and hence place it early in the sorted list of answers.

2.4 QA systems for non Indian languages

Since 1960, QA systems are being build and present systems are even concentrating on addressing visual QA. Considering the popularity of the European languages, particularly in English, many QA systems have been developed over time. Even the two most widely known QA systems (namely BASEBALL and LUNAR) were developed in English. Other QA systems in different European languages other than English follow the footsteps of the QA systems which were developed in English. Eventually, QA systems have been developed in other European languages such as Spanish, Italian, German, etc.

The mostly known QA evaluation track, i.e., TREC was first organized in 1992 and since then it is being conducted each year. In 2016, it celebrated its 25th years of contribution to the QA research. Recently, Spanish is added as non-English language. Each year, the TREC contributes a test set of documents and questions. As a result of that, the corpus for English QA research is increasing each year. Therefore, more than 25 years, English language gets the leverage of the TREC and it leaves the other languages behind.

The CLEF bridges this gap by initiating its own multilingual QA track. CLEF covers popular European languages such as Spanish, German, French, etc.

The followings are some example of notable QA systems in European languages other than English.

- **Spanish QA Systems:** Zheng (2002a), Vicedo et al. (2003),Roger et al. (2005), Pérez-Coutiño et al. (2005)
- **German QA Systems:** Zheng (2002a),Dong et al. (2011)
- **French QA Systems:** QRISTAL, Zheng (2002a)

- **Italian QA Systems:** Zheng (2002a)
- **Portuguese QA Systems:** Amaral et al. (2005)

Asia is the largest as well as the most populous continent that used to have millions of speakers of diverse languages. Like the European languages, research on QA is one of the major research topics of NLP. A number of research are being carried out in various Asian languages. Each language has its own challenges and history for developing towards the QA systems. Within the scope of this thesis, it is not possible to cover all such issues. Therefore, a few research on popular Asian languages are given below:

- **Arabic QA Systems:** Mohammed et al. (1993), Rosso et al. (2006), Benajiba et al. (2007), Kanaan et al. (2009), Trigui et al. (2010), Abouenour et al. (2012), etc.
- **Chinese QA Systems:** Yongkui et al. (2003), Sun et al. (2008), etc.
- **Japanese QA Systems:** Sakai et al. (2004), Isozaki et al. (2005), etc.

2.5 QA Systems for Indian Languages

Although research on QA was initiated beforehand, the scenario is different for Indian languages. In India, the QA research is in the nascent stage. India is a multicultural and multilingual country where there are 22 official languages. In spite of the fact that information seekers accept QA systems as a good alternative of search engines, in many Indian languages the research in QA has not been started yet.

Sahu et al. (2012) proposed a factoid Hindi QA system named 'Prashnottar'. The 'Prashnottar' applied handcrafted rules to identify question patterns and it can answer to the questions of type 'when', 'where', 'how many' and 'what time'. The reported accuracy of the system is around 68%.

Recently, Nanda et al. (2016) developed a QA system in Hindi. The QA system used machine learning technique to predict the type of the entity from the user question. They reported that the QA system was tested on 75 questions. Ray et al. (2018) criticize that the description of the document set is not given and it is not clear how and from where the system is extracting the answer. However, Nanda et al. (2016) reported the accuracy of the QA system is 90%.

A few Hindi-English cross lingual (c.f. 2.1.3) QA systems were reported. Larkey et al. (2003) proposed a English-Hindi cross lingual system that employed NLP techniques such as normalization, stop-word removal, transliteration, structured query translation, and language modeling using a probabilistic dictionary derived from a parallel

corpus. The work reported the challenges of building Hindi-English cross lingual system which includes proprietary encoding schemes of the web content, scarcity of Hindi-English parallel corpus. The system was tested on 15 queries and 41697 Hindi documents which were collected from BBC. The reported accuracy is 0.4298 in terms of mean average precision.

In 2003, Sekine and Grishman (2003) developed cross lingual QA system for Hindi and English in the framework of the TIDES program. They built the system within one month and argued that basic system can be constructed quickly once if other linguistic tools become available. The QA system takes questions written in English as input. It finds the candidate answers from Hindi BBC news articles. The corpus was collected during January to June 2003 (6 months) and it consisted of 5,557 news articles. The QA system consisted of four components: question examiner (QE), cross lingual IR (CLIR) system, answer finder (AF) and machine translation (MT) system. The QE identified the expected types of answer and the keywords. Then, the CLIR system used the keyword to retrieve relevant news articles. Next, AF extracted the candidate answers with confidence scores. Finally, the answer and context of the answer were translated back to English. An web-based interface was developed for accessing this QA system. The system was tested with 56 questions. The reported performance of the QA system was 0.25 in terms of MRR.

A multilingual (c.f. 2.1.3) restricted domain QA system was reported by Shukla et al. (2004). Universal Networking Language (UNL) (Uchida and Zhu, 2001) was used to convert contents of a document in Hindi or English to intermediate language. Usually, the question focuses and the expected answer type was determined by analyzing the user's query. For each question, an answer template was generated which was again converted to UNL expression. The template was used to match with the UNL expression for the documents. Finally, the matched answers were extracted from the UNL to natural language. The reported accuracy of this system was 60%. However, Ray et al. (2018) criticize that the authors did not report the corpus details such as number of questions and documents used in testing.

Kumar et al. (2003) developed a QA system for E-Learning Hindi Documents. Based on the keywords, they classified the questions into six categories. After removing the stopwords from the question, selected keywords were stemmed to find semantically equivalent words for query expansion using a self-constructed small lexical database. Thus, a user query was reformulated and fed into the retrieval engine. For answer extraction similarity heuristic was employed. The system was evaluated using a set of 60 questions from agriculture and science domain. The work claimed the system answered 86.67% of the questions.

For Telugu, Reddy and Bandyopadhyay (2006) proposed a dialogue based QA system for the railway domain. The core part of the QA system was dialogue manager

that was responsible for the flow of dialogues. The dialogue manager was also responsible for the coordination of the other components in the system. The dialogue manager acted as an intermediary between the user and the system. Query analyzer analyzed the user query and it was responsible for the generation of the tokens and keywords with the use of knowledge base. Based upon the keywords and tokens which were presented in the knowledge base, an appropriate frame was selected. A SQL statements were generated from the tokens. There were two main issues in the design of the railway information system, how to design railway database and knowledge base. The railway database contained the information about the arrival and departure time of each train and the information regarding fares. The knowledge base contained the tables like train name and station name. It also contained alias tables for station name and train name. Relational model was used to maintain the database tables. For each input query, the root words were identified by the query analyzer during query analysis. Based upon the detection of the keywords and tokens, a query frame was identified during query frame decision. After the generation of the query frame, SQL query was generated. Then the answer was retrieved from the database using SQL query.

Dhanjal et al. (2016) reported a QA system developed for Punjabi. The research work was based on the concept taken from physics: 'Point of Gravity'. In this approach the question and answer text were processed to extract numerical features so as to determine '*Point of Gravity*'. Matching Gravity Score values were computed for finding answer against a question query. The working principal of the system was based on the numerical features extracted from the given question and the respective comprehension. The features includes named entities, epistemic score (based on Punjabi language rules), lexical density, readability index, point of gravity and matching gravity score of the question as well as the sentence.

Bindu and Sumam Mary (2012) worked on developing a QA system for Malayalam. The author reported that the QA system was based on the named entity tagging and question classification. The useful information was extracted from the documents using document tagging. The Question classification module determined the type of the question. Various Machine Learning methods were used to tag the documents. Rule-Based Approach was used for Question Classification. The corpus was prepared by collecting documents from medical books, journals, and health magazines. Test questions were gathered from the native users and a total of 200 questions were tested. This work used precision and recall metrics for evaluation and the obtained values were 88.5% and 85.9% respectively.

2.6 Recent QA Research

Although the QA research had started since 1961, it is still an ongoing research issue. Nowadays, a few researchers look into the QA research from different perspectives. A few researchers concentrate to apply the QA research into machine comprehension text, i.e., answering questions after reading a short text or story. Recently, a number of training and evaluation datasets have been released like QuizBowl (Iyyer et al., 2014), CNN/Daily Mail based on news articles (Hermann et al., 2015), CBT based on children books (Hill et al., 2015), or SQuAD (Rajpurkar et al., 2016) and WikiReading (Hewlett et al., 2016), both based on Wikipedia. The emerging deep learning architectures like attention-based and memory augmented neural networks (Bahdanau et al., 2014; Graves et al., 2014) were applied successfully to address this research problem.

Another emerging research issue is the visual QA, i.e., answering natural language queries about images. In recent years, there has been immense progress in the fields of computer vision, object detection and NLP. The visual QA system is an extension of QA research that combines NLP with computer vision. Recently, research work (Cudic et al., 2018; Yu et al., 2018; Zhang et al., 2018) are being successfully carried out on a regular basis.

3

Corpus Acquisition

In computational linguistics, a corpus is a large and structured set of texts that can be used to perform statistical analysis and hypothesis testing, checking occurrences or validating linguistic rules within a specific language territory. The Brown corpus (Corpus, 1979), the first modern computerized corpus, which was created between 1961 and 1964 at Brown University, is considered as a significant landmark in corpus development. Succeeding the release of Brown corpus many noteworthy monolingual as well as multilingual corpora have been developed across many languages.
In the history of QA research, the first standard corpus for QA research (in English) was made available with the organization of TREC (1992). Since then new QA resources are being released in TREC each year. Since 2003, QA@CLEF has been providing monolingual and multilingual QA corpora. QA4MRE@CLEF also provides documents and multiple choice questions from various domains. NTCIR has been providing similar QA resources in Japanese since 1999. Besides the three aforesaid major contributors, some notable corpora development works (Cabrio et al., 2007; Inoue and Akagi, 2012; Louis and Nenkova, 2012; Verberne et al., 2006) on QA research can also be found in the literature.

3.1 Data Acquisition

Often the question arises in the context of corpus development that whether the web can be considered as a corpus. In this regard, two web-based approached to corpus development are (De Schryver, 2002): (i) web for corpus, in which the web is used as a source of texts in digital format for the subsequent implementation of an offline corpus; (ii) web as corpus, which uses the web directly as a corpus. A few researchers follow the web for corpus approach (Sinclair et al., 1987a; Sinclair, 1987b) whereas a few researchers (Bernardini et al., 2006; Fletcher, 2004; Kilgarriff and Grefenstette,

2003) followed the web as corpus approach. The practice of the web as a corpus for teaching and research has been proposed a number of times (Fletcher, 2004; Fletcher et al., 2001; Robb, 2003; Rundell, 2000). Particularly, the Wikipedia as web resource has become a promising corpus and a big frontier for researchers as it contains documents in various fields such as Arts, Geography, History, Science, Sports, etc. However, a considerable number of Wikipedia pages in Bengali are not properly created for a number of domains. In many cases the content of the Wiki-page is not present or the Wiki-page contains only a few sentences for majority of documents particularly in the history and agriculture domains. Furthermore, Bengali is a less computerized language. Therefore, the availability of authentic and well formatted documents is inadequate. This led us to manually prepare the corpus for a few domains. We manually prepared the data from authentic government text books of pre-college standard for the history and agriculture domains.

3.2 Document Acquisition

It has already been mentioned that the Wikipedia pages are inappropriate for a few domains in Bengali and for that reason we also use text books for creating question answering corpus in Bengali. However, text books are only available in hard copies. Therefore, we had to manually prepared the corpus from the text books. These difficulties restricted us in creating large collection of dataset. As stated earlier that the documents were not only collected from Wikipedia but also manually prepared from the text books. Documents were acquired for three domains, namely History, Agriculture and Geography. For the history domain, all the documents were prepared from authentic text books. Altogether 33 documents were prepared for the history domain. One of the prime occupations in India is agriculture. For the agriculture domain, half of the documents were collected from Wikipedia and the rest were manually prepared from authentic text books. A total of 4 documents was prepared for the agricultural domain of which 2 documents were obtained from Wikipedia and 2 documents from the text books. Although the size of the corpus for agricultural domain is tiny, it is very useful. However, all the 10 documents of the geography domain were acquired from Wikipedia. Thus, a total number of 47 documents were collected for corpus preparation. Out of the 47 documents, 35 were manually prepared from text books. Around 81 hours of rigorous manual typing work were performed to prepare soft copies of 35 documents. Table 3.2 presents the statistics of the acquired corpus.

3.3 Question Acquisition

The earlier section discusses document preparation for Bengali question answering research. However, question acquisition is a much more challenging task than the acquisition of text data. In the question answering corpus, usually the questions along with the answers (i.e., question-answer pairs) are bound to the associated documents. In this regard, the following issues were addressed:

- The answer to a question can be either directly extracted or inferred from the document.
- The question corpus size; i.e., the number of questions in the corpus, has to be reasonable.
- The prepared questions in the corpus should be diverse and of broad coverage.
- The task should involve as many question-setters as possible to reduce bias.

A cloud based service was developed to manage the question acquisition process. To involve a sizable number of question-setters, requests were sent to students for volunteer service. Our objective was to involve many volunteers to increase the question corpus size and to reduce bias. The students, who showed their interest by accepting the request, were provided the authenticity credentials. 40 students participated in the question corpus preparation. We formed two groups, namely Gr-A and Gr-B, each group was consisting of 20 students. The documents, which were collected in the document acquisition process, were circulated to each member of the group. Each member was assigned a document and they were asked to submit at least 10 questions for each document. Thus, we addressed the second and fourth issues.
We imposed constraints while questions were posed by the students. We did this to address the first and the third issue. While preparing the questions, the students had to follow the following constraints:

- The answer to the asked question must be present directly in the document or can be inferred from the source document; this addresses the first issue.
- Each volunteer must use at least 5 different interrogatives of their own choice; which addresses the fourth issue.

The documents along with the questions were stored in our web server. After collecting these questions in web server, the overlapping nature of the questions were analyzed. Hence, we introduce a new terminology, Question Overlapping Frequency

(QOF), which represents the number of members who formed exactly the same question text. For example, a question having QOF=5 implies that 5 members have posed the question. Subsequently, the students belonging to Gr-A were requested to answer the questions submitted by the students of Gr-B and vice versa. Thus, we collected the question-answer pairs and we retained only the questions having QOF value of >= 5; i.e., at least 5 students submitted the same question without any interaction among themselves. Setting a higher value of QOF increases the validity of the questions. Because it was observed from the submitted question set that the questions having low frequencies suffered from typographic errors and sometimes failed to satisfy the question submission constraints. However, this was not satisfied for all the cases where the QOF frequency was low for a question. There is another possibility that the questions with lower QOF value might not be erroneous, but rather rare and informative ones. Considering the importance of the questions with the lowest QOF value (i.e., QOF = 1, which implies that only one of the students submitted the question) we did not discard those questions. To deal with this situation, we sorted out the questions with QOF=1 and the validity of such questions was verified with human intervention. We found 12 such questions with QOF=1 and only 2 of them were identified as valid questions with the help of human intervention. Thus, we collected 2033 valid question-answer pairs.

3.4 Proposed Template

Nowadays, the use of *eXtensible Markup Language* (XML) is very popular to store data. XML is a markup language much like HTML and it was developed by *World Wide Web Consortium*[1] (W3C). XML was designed in such a way that it ~~was~~ is human-readable as well as machine-readable.

Table 3.1: Corpus tagset

Tag	Definition	Tag	Definition
Doc ID	Document ID number	Domain	Domain of the document
Lang	Language of the document	Title	Topic name
Text	Text part of the topic	Paragraph	Paragraph of text part
Questions	Questions answering part	Question	Single question information
Qid	Question ID number	Qtype	Question class
Qstr	Question	EAT	Expected answer type
QTT	Question topical target	Ans	Answer of the question

Hence, we chose the XML technology for corpus management and storing because of its popularity and ease of understanding. We also defined our own tagset to

[1] https://www.w3.org/

represent the corpus in XML format. The proposed tagset is given in Table 3.1. The tags in the tagset were used for three purposes - *document information*, *text annotation* and *question answering annotation*. The *document information* portion contains the unique document identification number, domain, script and the title of the document. The *text annotation* portion follows the *document information* portion. The *text annotation* portion contains the paragraphs which is used to answer the questions. The *text annotation* portion precedes the *question answering annotation* portion. The question-answer pairs are annotated in the *question answering annotation* portion.

We also proposed a template for representing each document. The template is depicted in Figure 3.1.

Figure 3.1: Document template

3.5 Question Answering Annotation

In this section, we describe the question answering based annotations. The question answering annotation deals with the annotation of questions along with the question analysis. Next, we describe the annotation in particular the Question Type, the Expected Answer Type, and the Question Topical Target.

3.5.1 Question Type

The set of question categories (classes) is referred to as question type taxonomy or question ontology. Although different question taxonomies have been proposed for different languages, no taxonomy existed for Bengali questions prior to the research reported in this thesis. For Bengali questions, we proposed a taxonomy which is described in Section 4.2. The proposed taxonomy is the only existing standard taxonomy for Bengali till date. Therefore, for Question Type (Qtype) annotation task, we followed the aforesaid taxonomy which is a single layer taxonomy with nine coarse-grained classes: PER (person), ORG (organization), LOC (location), TEM (temporal), NUM (number), METH (method), REA (reason), DEF (definition) and MISC (miscellaneous). However, the Qtype annotation is not straightforward for Bengali. The Bengali questions may take interrogatives of three categories: Unit Interrogative (UI), Dual Interrogative (DI), and Compound Interrogative (CI). The Qtype annotation of a UI question is straightforward and the possible value is the one of the 9 classes. Bengali interrogatives are discussed in details in Section 4.1. The example below shows the UI Qtype.

Question: <Qstr> সিন্ধু[1] সভ্যতা[2] কে[3] আবিষ্কার[4] করেন[5] ?[6] </Qstr>

[Sindh[1] civilization[2] who[3] discover[4] did[5] ?[6]]

<Qtype>PER</Qtype>

Gloss: Who discovered Sindh civilization?

Each DI is formed by two UIs. However, since both of the UIs take the same Qtype value, therefore for simplicity the Qtype value is annotated by a single value. The following example demonstrates the DI Qtype.

Question: <Qstr> কোন[1] কোন[2] অঞ্চল[3] বাবরের[4] আধিকারে[5] আসে[6] ?[7] </Qstr>

[which[1] which[2] area[3] Babar[4] control[5] come[6] ?[7]]

<Qtype>LOC</Qtype>

Gloss: Which were the areas that came under Babar's control?

A CI is formed by combining two different interrogatives. For each interrogative, a Qtype value is required. Therefore, a total of two Qtype values are required for annotation. The example given below shows the CI QType.

Question: <Qstr> কে[1] কবে[2] সিন্ধু[3] সভ্যতা[4] আবিষ্কার[5] করেন[6] ?[7] </Qstr>

[who[1] when[2] Sindh[3] civilization[4] discover[5] did[6] ?[7]]

<Qtype>PER-TEM</Qtype>
Gloss: Who and when discovered Sindh civilization?

3.5.2 Expected Answer Type

Prager et al. (2007) defined the Expected Answer Type (EAT) as the class of object (or rhetorical type of sentence) required by the question; in other words, it is the semantic category associated with the desired answer, chosen from a predefined set of labels. The named entity (NE) category of the answer to the question can be used as EAT. The tagset defined in IJCNLP-08 NERSSEAL shared task (Singh, 2008b) was used as the NER tagset. However, the EAT is not applicable (NA) when the Qtype is of type METH, REA and DEF; the reason being the answer is not an NE in such cases. For other types of Qtype, i.e., PER, ORG, LOC, TEM, NUM, and MISC, the answer is a single word or a noun phrase and the NE of the answer is the EAT.
Question: <Qstr> কোন[1] কোন[2] অঞ্চল[3] বাবরের[4] আধিকারে[5] আসে[6] ?[7] </Qstr>
[which[1] which[2] area[3] Babar[4] control[5] come[6] ?[7]]
<Qtype>LOC</Qtype>
<EAT>NEL</EAT> (i.e., named entity location, c.f. 5.2)
Gloss: Which were the areas that came under Babar's control?
The above example shows the EAT for QType LOC. The example given below shows the EAT for QType DEF.
Question: <Qstr> বেদ[1] কি[2] ?[3] </Qstr>
[Beda[1] what[2] ?[3]]
<Qtype>DEF</Qtype>
<EAT>NA</EAT>
Gloss: What is Beda?

3.5.3 Question Topical Target

Question Topical Target (QTT) is a part of the question that describes the entity about which the information request has been made. The QTT is sometimes referred to as question focus, or question topic. QTT corresponds to a noun or a noun phrase that is likely to be present in the answer. However, unlike English questions, the interrogatives of a Bengali question can appear at the beginning, within or at the end of the question text and it is rather difficult to separate the noun phrase when the interrogative appears within the question text. To follow the same strategy for all cases, the named entities in the question text are considered as QTT. Therefore, in the present work, QTT is represented as a comma separated named entity list.
Question: <Qstr> সূর্যের[1] ভর[2] কত[3] ?[4] </Qstr>
[Sun[1] mass[2] what[3] ?[4]]

<QTT> সূর্য </QTT>
Gloss: What is the mass of the Sun?
The above example shows the QTT with single NE. The example given below shows the QTT with multiple NEs.
Question: <Qstr> সূর্যকে[1] প্রদক্ষিন[2] করতে[3] পৃথিবীর[4] কত[5] সময়[6] লাগে[7] ?[8] </Qstr>
[Sun[1] orbit[2] do[3] Earth[4] how[5] time[6] require[7] ?[8]]
<QTT> সূর্য, পৃথিবী </QTT>
Gloss: How long does it take for the Earth to orbit the Sun?

3.6 Annotation Agreement

We hired six native (i.e., Bengali) language specialists for annotating the question answering portion. In order to minimize disagreement, six language specialists gathered to discuss the question type, EAT and QTT in details before initiating the annotation task. The annotators divided the whole annotation task in three subparts, namely QType annotation, EAT annotation and QTT annotation. Each subpart was evaluated by two independent language specialists. We measured the Inter-annotator agreement using non-weighted kappa coefficients (ref). The kappa coefficient for Qtype, EAT and QTT annotation tasks are 0.95, 0.91 and 0.89 respectively, which represent very high agreement(s).

3.7 Corpus Statistics

Annotation and distribution of questions in the corpus for three different domains, namely history, geography and agriculture, are given in Table 3.2. Also Table 3.2 details the document source and statistics for individual domains.

Table 3.2: Questions statistics

Domain	#Documents	Source	#Questions
History	33	Books	1922
Geography	10	Wikipedia	251
Agriculture	4	Wikipedia (2) + Books (2)	84

The distribution of different question types for individual domains is shown in Table 3.3. It was quite obvious that the presence of Person question type was the maximum in the History domain. However, any presence was not observed for person type question in the agricultural domain. In the corpus, a total of 2,257 questions were annotated according to three question answering based levels, namely Question Class, Expected Answer Type and Question Topical Target. The corpus statistics show that questions of type person (799 questions) appear more frequently than questions belonging to the other types. Also it is evident from the Table 3.3

Table 3.3: Qtype statistics

Type	**History**	**Geography**	**Agriculture**	**Overall**
Person	777	22	0	799
Organization	54	4	6	64
Location	137	6	11	154
Temporal	171	47	13	231
Numerical	35	60	15	110
Methodical	139	3	10	152
Reason	128	17	8	153
Definition	28	17	7	52
Miscellaneous	430	65	9	504
Person-Temporal	8	3	0	11
Person-Location	6	0	0	6
Temporal-Location	4	4	3	11
Numerical-Miscellaneous	5	3	2	10

that the compound question types are less frequent than the simple interrogative (discussed in Section 4.1) question types.

4

Question Analysis

4.1 Interrogatives in Bengali

Interrogative words or question words are used to form the questions. In English, nearly all question words start with the same character bigrams – WH (e.g., who, what, which, where, when, whom). Therefore, the question words are also called WH-words. Generally, the question type may be determined by the interrogative present in the question, such as the word 'why' in 'Why are you late?' describes that someone asks the reason for being late. However, all the question types can not be determined only by the interrogatives. In Bengali, the interrogative words not only describe important information about the expected answer but also indicate the Number representations, i.e.- singular or plural. The Bengali Interrogatives, which are identified by us, are shown in Table 4.1. Unlike English, there are many interrogative words present in the Bengali language. After analyzing the corpus, we have classified the interrogatives in three categories:

- Simple Interrogative(SI) or Unit Interrogative(UI)
- Dual Interrogative(DI)
- Compound/Composite Interrogative(CI)

4.1.1 Simple Interrogatives or Unit Interrogatives

The interrogative of this type is made up of a single interrogative word which can be considered as an Interrogative unit. Further, an SI can be classified into two subtypes according to answer indication number representation. An SI can indicate a Singular Answer (SA) or a Plural Answer (PA). If an SI indicates an SA, it is considered Singular Simple Interrogative (SSI) or Singular Unit Interrogative (SUI).

Table 4.1: Bengali Interrogatives

Sl.No	Interrogative	Category	Number Representation
1	(ke)	SSI	Singular
2	(kake)	SSI	Singular
3	(kahake)	SSI	Singular
4	(ke ke)	PDI	Plural
5	(kara)	PSI	Plural
6	(kar)	SSI	Singular
7	(kar kar)	DI	Plural
8	(kader)	PSI	Plural
9	(kon)	BSI	Singular/Plural
10	(kon kon)	DI	Plural
11	(ki)	NSI	Neutral
12	(ki ki)	DI	Plural
13	(koto)	BSI	Singular/Plural
14	(koiti)	BSI	Singular/Plural
15	(kokhon)	NSI	Neutral
16	(kothai)	NSI	Singular
17	(kobe)	NSI	Neutral
18	(keno)	NSI	Neutral
19	(kivabe)	NSI	Neutral
20	(kemon)	NSI	Neutral
21	(ke kobe)	CI	Singular
22	(kara kobe)	CI	Plural
23	(ke kokhon)	CI	Singular
24	(ke kar)	CI	Singular
25	(kobe kar)	CI	Singular
26	(ke kon)	CI	Singular

Otherwise, if it indicates a PA, it is considered Plural Single Interrogative (PSI). Sometimes an SI can indicate both, i.e., SA and PA. Moreover, sometimes it plays a neutral role, i.e., it can be considered as BSI (both) and NSI (neutral). Therefore, with an SI all the answer indication number representations are possible. Hence, we found four subcategories of SI, i.e., SSI, PSI, BSI and NSI.
For example, SI/UI: (‘ke’), (‘kara’), (‘kader’), (‘kahake’).
SSI/SUI: (‘ke’), (‘kahake’)
PSI/PUI: (‘kara’), (‘kader’)
BSI: (‘kon’), (‘koto’), (‘koiti’)
NSI:(‘kivabe’),(‘keno’).

4.1.2 Dual Interrogatives

Usually, a dual interrogative (DI) is formed by using an SI/UI twice. Nevertheless, all the SI/UI cannot be used to form a DI. Typically, all the SSI/SUI can be used

twice in a question to make a DI. Unlike Bengali, generally the presence of DI is not observed in other languages. Examples of DI are given below:
DI: ('ke ke'); using SSI ('ke')
DI: ('kar kar'); using SSI ('kar')
DI: ('ki ki'); using SSI ('ki')
Although a DI is consists of one SSI twice, but each DI indicates Plural Answer (PA) only. Therefore, ('ke') indicates SA , but ('ke ke') indicates PA. This implies that all DIs are implicitly PA.

4.1.3 Compound or Composite Interrogatives

In DI, an SI is used twice to form the interrogate. However, two different SIs are used for forming each compound interrogative (CI). A CI is formed for getting multiple answers. Hence, it is difficult to categorize it into SA or PA. Therefore, further simplification is needed for this type of questions. We have identified six CIs from the corpus.
CI = {কে কবে, কারা কবে, কে কার, কবে কার, কে কখন, কে কোন}

4.2 Question Type Taxonomies

The set of question categories is referred to as question taxonomiy or question ontology. Although different question taxonomies have been proposed in different question classification research, most of the recent learning-based and hybrid approaches are based on two layer taxonomy proposed by (Li and Roth, 2002c). The taxonomy proposed by (Li and Roth, 2002c) consists of six course-grained classes. So far, the broadest question taxonomy has been proposed by Hermjakob (Hermjakob et al., 2002) which consists of 180 classes. Considering the fact that Bengali question classification is at the early stage of development, initially we proposed a single-layer (i.e., it has no fine-grained classes) taxonomy. The taxonomy consists of only eight coarse-grained classes. Till date, any researcher does not propose any question taxonomy in Bengali. Initially, we proposed a single-layer taxonomy for Bengali question classification. Later, based on the coarse-grained classes, we extended the single-layer taxonomy into a two-layer taxonomy. We incorporated 69 fine-grained question classes. Table 4.2 presents the proposed two-layer taxonomy.
The taxonomy proposed by (Li and Roth, 2002c) contains 6 coarse-grained classes: Abbreviation, Description, Entity, Human, Location, and Numeric. Abbreviation and Description classes of (Li and Roth, 2002c) are not present in Bengali taxonomy. Two coarse-grained classes of (Li and Roth, 2002c), namely, Entity and Human have resemblance with Miscellaneous and Person, respectively, in the Bengali taxonomy. While Location and Number classes are present in both the taxonomies, Organization and Method classes are not present in (Li and Roth, 2002c). In 2-layer Bengali

Table 4.2: Two-layer Bengali question taxonomies

Coarse-grained	Fine-grained
Person (PER)	GROUP, INDIVIDUAL, APPELLATION, INVENTOR/ DISCOVERER, POSITION, OTHER
Organization (ORG)	BANK, COMPANY, SPORT-TEAM, UNIVERSITY, OTHER
Location (LOC)	CITY, CONTINENT, COUNTRY, ISLAND, LAKE, MOUNTAIN, OCEAN, ADDRESS, RIVER, OTHER
Temporal (TEM)	DATE, TIME, YEAR, MONTH, WEEK, DAY, OTHER
Numerical (NUM)	AGE, AREA, COUNT, LENGTH, FREQUENCY, MONEY, PERCENT, PHONE-NUMBER, SPEED, WEIGHT, TEMPERATURE, OTHER
Method (METH)	NATURAL, ARTIFICIAL
Reason (REA)	INSTRUMENTAL, NON-INSTRUMENTAL
Definition (DEF)	ANIMAL, BODY, CREATION, CURRENCY, FOOD, INSTRUMENT, OTHER, PLANT, PRODUCT, SPORT, SYMBOL, TECHNIQUE, TERM, WORD
Miscellaneous (MISC)	COLOR, CURRENCY, ENTERTAINMENT, LANGUAGE, OTHER, VEHICLE, AFFAIR, DISEASE, PRESS, RELIGION

taxonomy, 15 fine-grained classes of (Li and Roth, 2002c) are not present, namely, abbreviation, expression, definition, description, manner, reason, event, letter, substance, title, description, state, code, distance, and order.

All the coarse-grained classes of (Shih et al., 2005) are present in the Bengali taxonomy. However, the Method class of the Bengali taxonomy is not present in Shih et al. (2005). The Artifact class of Shih et al. (2005) is similar to Definition and Miscellaneous of the Bengali taxonomy. In the 2-layer Bengali taxonomy, 9 fine-grained classes of Shih et al. (2005) are not included, namely, firstperson, planet, province, political system, substance, range, number, range, and order.

The 5 fine-grained classes are introduced in Bengali taxonomy which are not present in (Li and Roth, 2006) and Shih et al. (2005). These 5 classes are: AGE, NATURAL, ARTIFICIAL, INSTRUMENTAL, and NONINSTRUMENTAL. The NATURAL and ARTIFICIAL fine-grained classes belong to the coarse-grained Method class which is not present in Li and Roth (2006) and Shih et al. (2005). Similarly, INSTRUMENTAL, NON-INSTRUMENTAL fine-grained classes belong to the coarse-grained Reason class. Also, the Reason coarse-grained class is not present in Li and Roth (2006) and Shih et al. (2005). The AGE fine-grained class belong to the coarse grained class Numerical.

The taxonomies proposed in Li and Roth (2006) and Shih et al. (2005) did not deal with causal and procedural questions. The proposed Bengali 2-layer taxonomy is based on the only available Bengali QA dataset (Banerjee and Bandyopadhyay, 2012b) which contains causal and procedural questions. Therefore, the Bengali taxonomy contains question classes for the causal and procedural questions. A few

fine-grained classes of Li and Roth (2006) and Shih et al. (2005) are not included in the taxonomy since such questions are not present in the Bengali QA dataset. However, the proposed Bengali taxonomy is not final for Bengali QA task. Increasing the size of the said dataset is still in process. Therefore, it is expected that the missing fine-grained classes will be incorporated in the taxonomy in future.

4.3 Question Classification

4.3.1 Features for Question Classification

In the task of machine learning based QC, deciding the optimal set of features to train the classifiers is crucial. The features used for the QC task can be broadly categorized into three different types: lexical, syntactic and semantic features (Loni et al., 2011). In the present work, we also employed these three types of features suitable for the Bengali QC task.

Loni et al. (Loni et al., 2011) represented questions similar to document representation in the vector space model, i.e., a question is represented as a vector described by the words inside it. Therefore, a question Q_i can be represented as:

$Q_i = (W_{i1}, W_{i2}, W_{i3}, \ldots, W_{i(N-1)}, W_{iN})$,

where, W_{ik} = frequency of the term k in question Q_i and N = total number of terms in the vocabulary.

Due to the sparseness of the feature vector, only non-zero valued features are kept. Therefore, the size of the samples is quite small despite the huge size of the feature space. All lexical, syntactic and semantic features can be added to the feature space which expands the feature vector.

In the present study, the features employed for classifying questions (cf. Table 4.2) are described in the following subsections. In addition to the features used for the coarse-grained classification, fine-grained classification uses an additional feature, namely coarse-class, i.e. the label of the coarse-grained class.

4.3.2 Lexical Features

Lexical features (f_L) of a question are extracted from the words appearing in the question. Lexical features include interrogative-words, interrogative-word-positions, interrogative-type, question-length, end-marker and word-shape.

Interrogative-words and interrogative-word-positions: The interrogative-words (e.g., what, who, which etc.) of a question are important lexical features. They are often referred to as *wh*-words. Huang et al. (Huang et al., 2008, 2009) showed that considering question interrogative-word(s) as a feature can improve the performance of question classification task for English QA. Because of the relatively

free word-ordering in Bengali, interrogative-words might not always appear at the beginning of the sentence, as in English. Therefore, the position of the interrogative (wh) words along with the interrogative words themselves have been considered as the lexical features. The position value is based on the appearance of the interrogative word in the question text and it can have any of the three values namely, first, middle and last.

Interrogative-type: Unlike in English, there are many interrogatives present in the Bengali language. Twenty six Bengali interrogatives were reported in Banerjee and Bandyopadhyay (2012b). In the present work, the Bengali interrogative-type (wh-type) is considered as another lexical feature. In Banerjee and Bandyopadhyay (2012b), the authors concluded that Bengali interrogatives not only provide important information about the expected answers but also indicate the number information (i.e., singular vs plural). In Banerjee and Bandyopadhyay (2012b), wh-type was classified to three categories: Simple Interrogative (SI) or Unit Interrogative (UI), Dual Interrogative (UI) and Compound/Composite Interrogative (CI).

Question length: Blunsom et al. (Blunsom et al., 2006) introduced the length of a question as an important lexical feature which is simply the number of words in a question. We also considered this feature for the present study.

End marker: The end marker plays an important role in Bengali QC task. Bengali questions end with either '?' or '|'. It has been observed from the experimental corpus that if the end marker is '|' (similar to dot '.' in English), then the given question is a definition question.

Word shape: The word shape of each question word is considered as a feature. Word shapes refer to apparent properties of single words. Huang et al. (Huang et al., 2008) introduced five categories for word shapes: all digits, lower case, upper case, mixed and other. Word shape alone is not a good feature for QC, however, when it is combined with other kinds of features, it usually improves the accuracy of QC (Huang et al., 2008; Loni et al., 2011). Capitalization feature is not present in Bengali; so we have considered only the other three categories, i.e., all digits, mixed and other.

Example-1: *ke gOdZa prawiRTA karena ?* (কে গৌড় প্রতিষ্ঠা করেন?)

Gloss: Who established Goura?

[0]All the Bengali examples in this paper are written in WX (Diwakar et al., 2010) notation which is a transliteration scheme for representing Indian languages in ASCII.

Lexical features: wh-word: ke; wh-word position: first; wh-type: SI; question length: 5; end marker: ? word shape: other

4.3.3 Syntactic Features

Although different works extracted different syntactic features (f_S), the most commonly used f_S are Part of Speech (POS) tags and head words (Loni, 2011).

POS tags: In the present work, we used the POS tag of each word in a question such as NN (Noun), JJ (adjective), etc. POS of each question word is added to the feature vector. A similar approach was successfully used for English (Blunsom et al., 2006; Li and Roth, 2006). This feature space is sometimes referred to as the bag-of-POS tags (Loni et al., 2011). The Tagged-Unigram (TU) feature was formally introduced by (Loni et al., 2011). TU feature is simply the unigrams augmented with POS tags. Loni et al. (Loni et al., 2011) showed that considering the tagged-unigrams instead of normal unigrams can help the classifier to distinguish a word with different tags as two different features. For extracting the POS tags, the proposed classification work in Bengali uses a *Bengali Shallow Parser*[1] which produces POS tagged data as intermediate result.

Question head word: Question head-word is the most informative word in a question as it specifies the object the question is looking for (Huang et al., 2008). Correctly identifying head-words can significantly improve the classification accuracy. For example, in the question "*What is the oldest city in Canada?*" the headword is 'city'. The word 'city' in this question can highly contribute to classify this question as LOC: city.
Identifying the question's head-word is very challenging in Bengali because of its syntactic nature and no research has been conducted so far on this. Based on the position of the interrogative in the question, we use heuristics to identify the question head-words. According to the position of the interrogative, three cases are possible.
Position-I (at the beginning): If the question-word (i.e., marked by WQ tag) appears at the beginning then the first NP chunk after the interrogative-word is considered as the head-word of the question. Let us consider the following question.
Example-2: *ke*(/WQ) *gOdZa*(/NNP) *prawiRTA*(/NN) *karena*(/VM) ?(/SYM) (কে গৌড় প্রতিষ্ঠা করেন?)
English Gloss: Who established Goura ?
In the above example, gOdZa is the head-word.

[1] http:// ltrc.iiit.ac.in/analyzer/bengali/

Position-II (in between): If the position of the question-word is neither at the beginning or at the end then the immediate NP-chunk before the interrogative-word is considered as the head-word. Let us consider the following question.
Example-3: *gOdZa*(/NNP) *koWAyZa*(/WQ) *abashiwa*(/JJ) ?(/SYM) (গৌড় কোথায় অবস্থিত?)
English Gloss: Where is Goura situated ?
In the above example gOdZa is considered as the question head-word.
Position-III (at the end): If the question-word appears at the end (i.e., just before the end of sentence marker) then the immediate NP-chunk before the interrogative-word is considered as the question head-word. Therefore, a similar action is taken for Position II and III.
Example-4:[*bAMlAxeSe arWanIwi kaleja*](/NNP) *kayZati* (/WQ) ?(/SYM) (বাংলাদেশে অর্থনীতি কলেজ কয়টি ?)
English Gloss: How many economics colleges are in Bangladesh?
Therefore, in the Example-4 [*bAMlAxeSe arWanIwi kaleja*] is the question head-word.
Now, if we consider the example "*ke gOdZa prawiRTA karena ?*" then the syntactic features will be: [{WQ, 1},{NNP, 1}, {NN, 1}, {VM, 1},{head-word,gOdZa}]. Here a feature is represented as {⟨ POS, frequency ⟩}.

4.3.4 Semantic Features

Semantic features (f_M) are extracted based on the semantics of the words in a question. In this study, related words and named entities are used as f_M.

Related word: A Bengali synonym dictionary is used to retrieve the related words. Three lists of related words were manually prepared by analyzing the training data.
date:{ *janmaxina, xina, xaSaka, GantA, sapwAha, mAsa, baCara, ...*,etc.};
food:{ *KAbAra, mACa, KAxya, mAKana, Pala,Alu, miRti, sbAxa, ...*, etc.};
human authority:{ *narapawi, rAjA, praXAnamanwrI, bicArapawi, mahAparicAlaka, ceyZAramyAna, jenArela, sulawAna, samrAta, mahAXyakRa, ...*, etc.};
If a question word belongs to any of the three lists (namely date, food, human activity), then its category name is added to the feature vector. For instance, the question "*ke gedZera sbAXIna narapawi Cilena ?*" (gloss: who was the independent ruler of Goura ?) contains the word *narapawi* which belongs to the human authority list. For this example question the semantic feature is added to the feature vector as: [{human-authority, 1}].

Named entities: We used named entities (NE) as a semantic feature which was also recommended in other works (Blunsom et al., 2006; Li and Roth, 2006) on

other languages. To identify the Bengali named entities in the question text, a Margin Infused Relaxed Algorithm (MIRA) based Named Entity Recognizer (NER) (Banerjee et al., 2014b) is used for the present study. For the question in Example-5, the NE semantic feature is added to the feature vector as: [{Location, 1}].
Example-5: *ke gOdZa*[Location] *prawiRTA karena*? (কে গৌড় প্রতিষ্ঠা করেন?)
English Gloss: Who established Goura ?

4.3.5 Coarse-grained Classification using Individual Classifiers

Many supervised learning approaches (Blunsom et al., 2006; Huang et al., 2008; Li and Roth, 2002b) have been proposed for QC over the years. But these approaches primarily differ in the classifier they use and the features they train their classifier(s) on (Loni, 2011). We assume that a Bengali question is unambiguous, i.e., a question belongs to only one class. Therefore, we considered multinomial classification which assigns the most likely class from the set of classes to a question. Recent studies (Huang et al., 2008; Silva et al., 2011; Zhang and Lee, 2003) also considered one label per question.
Initially, we employed four classifiers, namely Naïve Bayes (NB), Kernel Naïve Bayes (k-NB), Rule Induction (RI) and Decision Tree (DT) classifiers.
Classification accuracy is used to evaluate the results of our experiments. Accuracy is the widely used evaluation metric to determine the class discrimination ability of classifiers, and is calculated using the following equation.

$$\text{accuracy} = \frac{\text{number of correctly classified samples}}{\text{total number of tested samples}}$$

Naïve Bayes

Naïve Bayes (NB) classifier is a simple probabilistic classifier based on Bayes' theorem with naive independent assumptions; it assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature, given the class variable.
Using the simplest assumption of a constant prior distribution, Bayes theorem leads to a straightforward relationship between conditional probabilities. Given a class label C with m classes, $c_1, c_2, \ldots, c_m$, and an attribute vector x of all other attributes, the conditional probability of class label c_i can be expressed as follows:

$$P(C = c_i \mid x) = \frac{P(x \mid C = c_i) \times P(C = c_i)}{P(x)}$$

where $P(C = c_i)$ is the probability of class label c_i and can be estimated from the data directly. The probability of a particular unknown sample, P(x), does not have

to be calculated because it does not depend on the class label and the class with highest probability can be determined without its knowledge.

Kernel Naïve Bayes

Kernel Naïve Bayes (k-NB) classifier is the modified version of NB classifier that uses estimated kernel density. Conditional probability $P(C = c_i \mid x)$ can be written as a kernel density estimate for class c_i as follows:

$$P(C = c_i \mid x) = f_i(x)$$

$$f_i(x) = \sum_{t=1}^{n} K_i(x, x_i)$$

where, x_t are training points and $K_i(x, x_t)$ is a kernel function.

Rule Induction

Rule Induction (RI) learns a pruned set of rules with respect to information gain. It works similar to the propositional rule learner named Repeated Incremental Pruning to Produce Error Reduction (RIPPER, Cohen 1995). Starting with the less prevalent classes, the algorithm iteratively grows and prunes rules until there are no positive examples left or the error rate is greater than 50%.
In the growing phase, for each rule greedy conditions are added to the rule until the rule is perfect (i.e. 100% accurate). The procedure tries every possible value of each attribute and selects the condition with the highest information gain. In the pruning phase, for each rule any final sequence of the antecedents is pruned with the pruning metric $\frac{p}{(p+n)}$.

Decision Tree

Decision trees (DT) are powerful classification methods which often can also easily be understood. In order to classify an example, the tree is traversed top-down. Every node in a decision tree is labelled with an attribute. The example's value for this attribute determines which of the outgoing edges is taken. For nominal attributes, we have one outgoing edge per possible attribute value, and for numerical attributes the outgoing edges are labelled with disjoint ranges. This decision tree learner works similar to Quinlan's C4.5 or CART.

4.3.6 Dataset

We carried out our experiments on the dataset described in the earlier section. As discussed, the questions in this dataset are acquired from different domains, e.g.,

education, geography, history, science, etc. We divided the question corpus into 7:3 ratio for experimentation. The experimental dataset consists of 1100 Bengali questions of which 70% are used for training and the rest (330 questions, 30%) for testing the classification models.

Experiments

We employed four classifiers (i.e., NB, k-NB, RI and DT) and used well known, widely used *Rapid Miner* [2] tool for the experimentation. Primarily, we used the lexical features of the questions. NB was used as the Baseline system and it achieved classification accuracy of 80.65% with lexical features. It was noted from the experiments that performance of the baseline system drastically fall on ORG class (Precision-14.41%, Recall-41.18%) and MTHD class (Precision-34.62%, Recall-75.27%). Although k-NB classifier increased the accuracy, it failed to improve classification performance on ORG and MTHD classes. RI classifier increased the classification accuracy (83.31%) as well as the performance of ORG and MTHD classes. DT classifier performed the best among all the classifiers (accuracy 84.19%) and it exceptionally performed well on ORG, MTHD and others classes. The detailed results are shown in Table 4.3.
Then, we used both lexical and semantic features together and applied NB, k-NB, RI and DT classifiers individually. It was noted from the experimental results that inclusion of semantic features improves the performance of all the classifiers. The experimental results are illustrated in table 4.3.
Finally, we used all the features, i.e., lexical, syntactical and semantic features. It enhanced the classification performances of K-NB and NB classifiers on ORG and MTHD classes. With all the features, NB classifier outperformed DT classifier handling NUM classes and RI classifier outperformed DT classifier handling TEMP classes. However, overall DT classifier (accuracy 87.63%) performed well on classifying Bengali questions. The detailed results are shown in Table 4.3.

Table 4.3: Experimental results with individual classifiers

Classifier	f_L	f_L+f_S	f_L+f_S+f_M
NB	80.65	81.34	81.89
k-NB	81.09	82.37	83.21
RI	83.31	84.23	85.57
DT	84.19	85.69	87.63

[2]http://www/rapid-i.com

4.3.7 Learning Combined Classifiers

The following sections describe the state-of-the art classifier combination approaches used in our work.

Ensemble Learning

Two popular methods for creating accurate classifier ensembles are bagging (Breiman, 1996c) and boosting (Freund et al., 1996; Schapire, 1990). These methods rely on resampling techniques to obtain different training sets for each of the classifiers.

Bagging: Bagging or bootstrap aggregation is a machine learning ensemble meta-algorithm technique proposed by (Breiman, 1996a,b). It can be used to improve the classification in terms of stability and classification accuracy. It also reduces variance and helps avoid "overfitting". The ensemble bagging algorithm (Breiman, 1996a) is given in Algorithm 1.

Algorithm 1: Bagging Algorithm

Input : Training set T of N examples,
Learning Model M (e.g., Decision Tree, Naïve Bayes, etc.),
Bagging size S

Output : ensemble model and predicted class

Training:

for $s = 1 \ldots S$ **do**
  Randomly sample N examples with replacement from the training set T and generate training set T_K
  Apply base model M on training set T_K to generate model M_S
end

Classification:

for *each model of* M_S **do**
  Predict class label via majority voting
  Return the class label that has been predicted most often
end

Although it is usually applied to decision tree models, it can be used with any type of model. Bagging trains a number of base learners by bootstrap sampling to get an aggregated prediction. Each classifier's training set T_K is generated by randomly selecting with replacement from N examples, where N is the size of the original training set. In the resulting training set T_K, some of the original examples may be repeated. Thus, each individual classifier B_L in the ensemble is generated with a different random sampling of the original training set T.

Boosting: Boosting is another well-known machine learning ensemble meta-algorithm technique. The ensemble boosting algorithm (Freund et al., 1996) is given in Algorithm 2.

Algorithm 2: Boosting Algorithm

Input : Training set T of N example,
Learning Model M (e.g., Decision Tree, Naïve Bayes, etc.),
Number of steps S

Output : ensemble model and predicted class

Training:

$D_t(1) = \frac{1}{N}$

for $t = 1 \ldots S$ **do**

Take K samples from the training set according to D_t

Train a classifier h_t on the samples. Calculate the error ε_t of h_t :

$$\varepsilon_t = \sum_{i:h_t(x_i)\neq y_i} D_t$$

Calculate weight β_t of h_t:

$$\beta_t = \frac{\varepsilon_t}{(1-\varepsilon_t)}$$

Calculate new sampling distribution:

$$D_{t+1}(1) = \frac{D_t(i)}{Z_t} \times \begin{cases} \beta_t & \text{if } h_t(x_i) = y_i \\ 1 & \text{otherwise} \end{cases}$$

Weight w_t of classifier h_t:

$$w_t = \log(\frac{1}{\beta_t})$$

end

Classification:

Classify according to the weighted majority of classifiers

Return the class that corresponds to the maximal sum of weights

This ensemble method produces a series of classifiers and the training set used by a classifier model is based on the performance of the earlier classifier model. In boosting, examples that are incorrectly predicted by previous classifiers in the series are chosen more often than the examples that are correctly predicted. Boosting attempts to produce new better classifiers by selecting incorrectly predicted samples than the correctly predicted samples from the training set used by the previous classifier. Freund et al. (1996)) developed a famous boosting algorithm: AdaBoost. There are two versions of AdaBoost: AdaBoost.M1 and AdaBoost.M2. In the present work, AdaBoost.M1 was used as the boosting method. The selected boosting method gives each training example equal weight. Therefore, if the training size consists of N ex-

amples then each example gets a weight of $\frac{1}{N}$. D_t is the sampling distribution, where $D_t(i)$ represents a probability that example i from the training dataset gets selected. A sample distribution D_t for building the j^{th} model is constructed by modifying the sampling distribution D_{t-1} from the $(j-1)^{th}$ step. Examples classified incorrectly in the previous step receive higher weights in the new data to cover incorrectly classified samples.

Stacking

Stacking or stacked generalization is another approach to classifier combination. In ensemble approaches (namely, bagging and boosting), multiple models of the same classifiers are used. However, in stacking, multiple models are developed by using different classifiers. In this study, we used the implementation of (Wolpert, 1992) for the stacking experiments. The stacking algorithm is given in Algorithm 3.

Algorithm 3: Stacking Algorithm

Input : Training set T of N example,
Set of classifiers for base and model learners

Output : Combined model and predicted class

Training:

Step 1: Split the training set T into two disjoint sets T_a and T_b

Step 2: Train base learners $C_1, C_2, \ldots, C_{n-1}, C_n$ on the first training set T_a

Step 3: Test the base learners $C_1, C_2, \ldots, C_{n-1}, C_n$ on the second training set T_b

Step 4: Using the predictions from Step 3 as the inputs, and the correct responses as the outputs, train a higher level learner

Voting

In this study, we used majority voting and the procedure is given in Algorithm 4. As the name suggests, this voting system selects the classification class which is proposed (i.e., voted) by majority of the base models.

4.3.8 Coarse-grained Classification using Classifiers Combination

The empirical study of state-of-the-art classifier combination approaches (i.e., ensemble, stacking, and voting) was performed. We used two contemporary methods for creating accurate ensembles, namely, bagging and boosting. We employed the Rapid Miner tool for all the experiments reported here. Each of the three classifier combination approaches was tested with NB, k-NB, RI and DT classifiers. Each experiment can be thought of as a combination of three experiments since each

Algorithm 4: Voting Algorithm

Input : Training set T of N example, m classification classes $\{C_1, C_2, \ldots, C_m\}$ and a set of base learners $M = \{M_1, M_2, \ldots, M_K\}$

Output : Predicted class

Training:

Train base learners on the training set T of N examples.
All the models are assigned same voting weight.

Classification:

for *each classification class* c_i **do**
  Count the votes received by this hypothesis from the individual classifiers
end
The class which receives the maximum votes is selected as the majority decision

classifier model was tested on $\{f_L\}$, $\{f_L, f_S\}$ and $\{f_L, f_S, f_M\}$ feature sets separately. Overall thirteen experiments were performed for coarse-grained classification and the evaluation results are reported in Table 4.4.

Table 4.4: Classifier combination results for coarse-grained classification

Approach	**Base-Learner**	**Model-Learner**	f_L	$f_L + f_S$	$f_L + f_S + f_M$
Bagging	NB	x	81.53	82.77	83.25
	k-NB	x	82.09	83.37	84.22
	RI	x	83.96	85.61	86.90
	DT	x	85.23	86.41	**91.27**
Boosting	NB	x	81.74	82.71	83.51
	k-NB	x	83.86	85.63	86.87
	RI	x	83.55	85.59	86.27
	DT	x	85.21	86.58	**91.13**
Stacking	k-NB, RI, DT	NB	81.76	82.79	83.64
	NB, RI, DT	k-NB	83.86	85.54	86.75
	NB, k-NB, DT	RI	85.55	87.69	**91.32**
	NB, k-NB, RI,	DT	85.07	86.73	89.13
Voting	NB, k-NB, RI, DT	x	86.59	88.43	**91.65**

Ensemble Bagging

The bagging approach was applied separately to four classifiers (i.e., NB, k-NB, RI and DT) and the obtained accuracies are reported in Table 4.4. Initially, the size (i.e., number of iterations) of the base learner was set to 2. Subsequently, experiments were performed with gradually increasing size ($size > 2$). The classification accuracy enhanced with increase in size. However, after a certain size, the accuracy was almost stable. At $size = 2$ and feature set $\{f_L, f_S, f_M\}$, the NB classifier achieved 82.23%

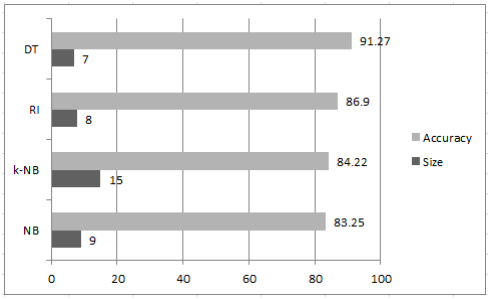

Figure 4.1: Size and accuracy variation in Bagging with $\{f_L, f_S, f_M\}$

accuracy and at $size \geq 9$, it became stable with 83.25% accuracy. At $size = 2$ and feature set $\{f_L, f_S, f_M\}$,the k-NB classifier achieved 83.87% accuracy and at $size \geq 15$, it became stable with 84.22% accuracy. At $size = 2$ and feature set $\{f_L, f_S, f_M\}$, the RI classifier achieved 85.97% accuracy and at $size \geq 8$, it became stable with 86.90% accuracy. At $size = 2$ and feature set $\{f_L, f_S, f_M\}$,the DT classifier achieved 88.09% accuracy and at $size \geq 7$, it became stable with 91.27% accuracy. It was observed from the experiments that with bagging the DT classifier performs the best on any feature set for any size. For the experiments with the f_L features, the bagging size of NB, k-NB, RI and DT are 12, 19, 11 and 10 respectively after which classification accuracy becomes stable. Similarly, for experiments with $\{f_L, f_S\}$ feature set, the optimal bagging sizes are 10, 17, 9 and 8 for NB, k-NB, RI and DT respectively after which the corresponding classification accuracies converge. Figure 4.1 shows the variation in size and accuracy for the best feature set.

Ensemble Boosting

Like bagging, boosting (AdaBoost.M1) was also applied separately to the four base classifiers. Table 4.4 tabulates the accuracies obtained with the boosting approach with the four classifiers. Here, we empirically fixed the iterations of boosting for the four classifiers to 12, 16, 10 and 8 respectively for the feature set $\{f_L, f_S, f_M\}$, since the corresponding weight of $\frac{1}{\beta_t}$ becomes less than 1 beyond those values. If $\frac{1}{\beta_t}$ is less than 1, then the weight of the classifier model in boosting may be less than zero

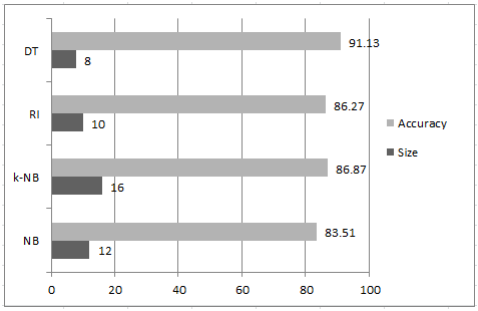

Figure 4.2: Size and accuracy variation in Boosting with $\{f_L, f_S, f_M\}$

for that iteration. Figure 4.2 shows the variation in size and accuracy for the best feature set.
Similarly, for the feature sets $\{f_L, f_S\}$ and $\{f_L\}$ the iterations are set to 13, 18, 12, 9 and 14, 19, 14, 11 respectively for the four classifiers. Overall the DT classifier performs the best. However, unlike in bagging, k-NB performs better than RI with boosting.

Stacking

In stacking, three out of the four classifiers are used as the base learners (BL) and the remaining classifier is used as the model learner (ML). Therefore, four experiments were conducted separately for each of the four classifiers as the ML. The obtained accuracies are reported in Table 4.4.
Experimental results revealed that stacking with RI as the model learner and NB, k-NB, DT as the base learners achieves the best classification accuracy.

Voting

In voting, four classifiers altogether were used as the base learners and majority vote was used as the voting approach. The evaluation results of the voting approach are presented in Table 4.4.

Observations on Coarse-Grained Classification

The automated Bengali QC system using single classifier approach was based on four classifiers, namely NB, k-NB, RI and DT, which were used separately. The experimental results obtained in single classifier approach are shown in Table 4.3. In that work, NB was used as the baseline and the DT classifier achieved the highest accuracy of 87.63% (cf. Table 4.4). A comparison of the results in Table 4.3 and Table 4.4 reveals that each classifier combination model performs better than the individual single classifier models in terms of classification accuracy.
In comparison to the earlier experiments using single classifier approach, classification accuracy of each classifier increases notably with bagging approach. The classification accuracy on the $\{f_L\}$, $\{f_L, f_S\}$ and $\{f_L, f_S, f_M\}$ feature sets increases by 1.04%, 0.72% and 3.64% for the best performing DT classifier. Similarly, with the boosting approach, the classification accuracy for the best performing DT classifier notably increases by 1.02%, 0.89% and 3.50% on $\{f_L\}$, $\{f_L, f_S\}$ and $\{f_L, f_S, f_M\}$ feature sets respectively. The stacking approach increases the accuracy on the $\{f_L, f_S\}$ feature set than the bagging and boosting approaches. This approach increases the classification accuracy by 1.36%, 2.74% and 0.69% on the $\{f_L\}$, $\{f_L, f_S\}$ and $\{f_L, f_S, f_M\}$ feature sets respectively. The voting approach not only increases the classification

accuracy, but also provides the maximum accuracy for all the feature sets than the other combination approaches. The voting approach increases the classification accuracy on the $\{f_L\}$, $\{f_L, f_S\}$ and $\{f_L, f_S, f_M\}$ feature sets by 2.40%, 2.40% and 4.02% respectively. Therefore, overall the voting approach with majority voting performed the best among the four classifier combination approaches.

Table 4.5: Accuracies obtain on the fine-grained classification using individual classifiers

Classifier	Class	f_L	f_L+f_S	f_L+f_S+f_M
NB	F_{PER}	74.07	75.54	77.07
	F_{ORG}	75.33	76.55	77.70
	F_{LOC}	76.15	77.02	77.87
	F_{TEM}	75.74	77.16	77.97
	F_{NUM}	74.61	75.45	76.55
	F_{METH}	76.35	77.42	78.50
	F_{REA}	76.19	77.20	78.02
	F_{DEF}	76.30	77.45	78.56
	F_{MISC}	75.80	76.95	77.40
k-NB	F_{PER}	75.72	77.33	78.41
	F_{ORG}	76.76	77.97	79.28
	F_{LOC}	77.52	78.55	79.40
	F_{TEM}	77.22	78.73	79.57
	F_{NUM}	76.09	76.94	78.05
	F_{METH}	77.92	79.14	80.24
	F_{REA}	77.82	79.36	80.33
	F_{DEF}	77.99	79.40	80.43
	F_{MISC}	77.37	78.74	79.60
RI	F_{PER}	77.96	79.04	80.12
	F_{ORG}	78.29	79.56	80.75
	F_{LOC}	77.67	78.36	79.18
	F_{TEM}	79.17	80.76	81.73
	F_{NUM}	78.04	79.03	80.42
	F_{METH}	79.87	81.00	82.12
	F_{REA}	79.62	80.93	82.06
	F_{DEF}	78.98	80.28	81.28
	F_{MISC}	78.59	79.91	80.90
DT	F_{PER}	80.37	82.06	83.61
	F_{ORG}	78.78	80.26	81.68
	F_{LOC}	78.51	79.63	80.94
	F_{TEM}	80.58	82.03	83.50
	F_{NUM}	79.00	80.50	81.85
	F_{METH}	80.62	82.55	84.47
	F_{REA}	80.51	82.49	84.42
	F_{DEF}	79.89	81.07	82.49
	F_{MISC}	79.74	81.72	84.07

4.3.9 Fine-grained Classification using Individual Classifiers

Initially, we applied NB, k-NB, RI and DT classifiers separately like the experiments of coarse-grained classification. Each classifier was trained with $\{f_L\}$, $\{f_L, f_S\}$ and $\{f_L, f_S, f_M\}$ feature sets. The performance of the classifiers increases gradually with incorporation of syntactic and semantic features (i.e., $\{f_L\} \rightarrow \{f_L, f_S\} \rightarrow \{f_L, f_S, f_M\}$). The NB classifiers achieved around 77% accuracy while the k-NB and RI classifiers achieved around 80% accuracy for the fine-grained question classes. Only the DT classifier obtained more than 80% accuracy for all the question classes. The detailed evaluation results for the fine-grained question classification task using individual classifiers are given in Table 4.5.

4.3.10 Fine-grained Classification using Classifiers Combination

The subsequent sections describe the experiments with classifier combination approaches.

Ensemble Bagging

In this approach, we use four classifiers as base learners individually: NB, k-NB, RI and DT. Initially, the base learners are trained using the lexical features (f_L). Then semantic and syntactic features are added gradually for classification model generation. Therefore, three classification models were generated for each base learner. Thus, altogether 12 models were prepared for bagging. Like coarse-grained clas-

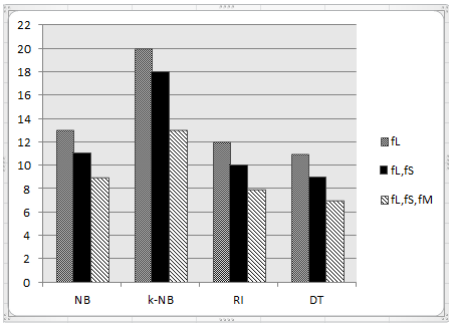

Figure 4.3: Size variation in Bagging

sification, initially the size (number of iteration) of the base learner was set to 2. Subsequently experiments were performed with gradually increasing sizes ($size > 2$). The classification accuracy increased with higher values of size. However, after certain iterations the accuracy was almost stable. For the fine-grained classes of PER coarse-class (i.e., F_{PER}), with $\{f_L, f_S, f_M\}$) feature set at $size = 2$, the NB classifier achieved 81.98% classification accuracy and at $size \geq 9$, it became stable with 82.87% accuracy. Similarly, with $\{f_L, f_S, f_M\}$ feature set the k-NB, RI and DT classifiers achieved stable accuracies at *size* equal to 13, 8 and 7 respectively. For the

lexical feature set, the bagging size of NB, k-NB, RI and DT were 13, 20, 12 and 11 respectively after which the classification accuracy became stable. For the combined lexical and syntactic features, the recorded bagging size of NB, k-NB, RI and DT were 11, 18, 10 and 9 respectively. Figure 4.3 depicts the iteration size for the bagging approach.

Table 4.6: Accuracies obtain on the ensemble results of fine-grained classification

		Bagging			**Boosting**		
		f_L	f_L+f_S	$f_L+f_S+f_M$	f_L	f_L+f_S	$f_L+f_S+f_M$
NB	F_{PER}	79.65	81.23	82.87	79.89	81.41	82.95
	F_{ORG}	81.01	82.32	83.55	81.65	82.73	83.98
	F_{LOC}	81.89	82.82	83.73	82.28	83.85	85.04
	F_{TEM}	81.45	82.97	83.84	81.89	83.01	83.97
	F_{NUM}	80.23	81.13	82.31	81.02	81.92	83.03
	F_{METH}	82.10	83.25	84.41	82.25	83.37	84.53
	F_{REA}	81.93	83.02	84.17	82.06	83.11	84.23
	F_{DEF}	82.05	83.29	84.47	82.09	83.32	84.56
	F_{MISC}	81.51	82.75	83.23	81.62	82.79	83.75
k-NB	F_{PER}	80.13	81.83	82.97	80.17	81.91	83.02
	F_{ORG}	81.23	82.51	83.89	81.29	82.63	83.91
	F_{LOC}	82.03	83.12	84.02	82.10	83.17	84.09
	F_{TEM}	81.71	83.31	84.20	81.79	83.39	84.28
	F_{NUM}	80.52	81.42	82.59	80.63	81.58	82.69
	F_{METH}	82.45	83.75	84.91	82.48	83.79	84.98
	F_{REA}	82.35	83.98	85.01	82.41	84.02	85.09
	F_{DEF}	82.53	84.02	85.11	82.61	84.12	85.13
	F_{MISC}	81.87	83.32	84.23	81.91	83.39	84.28
RI	F_{PER}	81.85	82.98	84.12	81.92	83.06	84.22
	F_{ORG}	82.19	83.53	84.78	82.25	83.61	84.85
	F_{LOC}	81.54	82.27	83.13	81.55	82.26	83.15
	F_{TEM}	83.12	84.79	85.81	83.18	84.85	85.93
	F_{NUM}	81.93	82.97	84.43	82.01	83.03	84.49
	F_{METH}	83.85	85.04	86.22	83.91	85.06	86.31
	F_{REA}	83.59	84.97	86.15	83.68	85.11	86.33
	F_{DEF}	82.92	84.28	85.33	82.95	84.32	85.41
	F_{MISC}	82.51	83.89	84.93	82.57	83.93	84.98
DT	F_{PER}	84.79	86.57	88.21	84.81	86.63	88.53
	F_{ORG}	83.11	84.67	86.17	83.14	84.73	86.23
	F_{LOC}	82.83	84.01	85.39	82.87	84.13	85.52
	F_{TEM}	85.01	86.54	88.09	85.03	86.58	88.15
	F_{NUM}	83.34	84.92	86.35	83.38	84.97	86.44
	F_{METH}	85.05	87.09	89.11	85.09	87.14	89.12
	F_{REA}	84.93	87.02	89.06	84.96	87.11	89.09
	F_{DEF}	84.28	85.53	87.02	84.29	85.55	87.05
	F_{MISC}	84.12	86.21	88.69	84.15	86.23	88.73

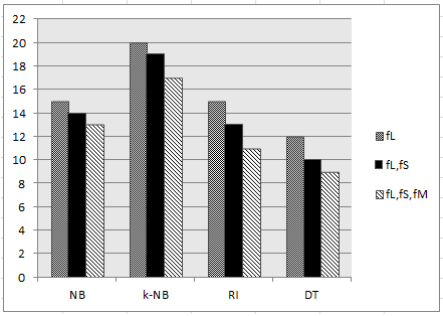

Figure 4.4: Size variation in Boosting

Ensemble Boosting

Like the ensemble bagging approach, we applied boosting (i.e., AdaBoost.M1) separately to the four classifiers. Experimental results confirm that performances of the four base classifiers improve slightly using AdaBoost.M1. Table 4.6 presents the results of the boosting experiments and shows that altogether DT outperforms the other classifiers in the ensemble approach, i.e., bagging and boosting.
In the boosting approach, the number of iterations depends on $\frac{1}{\beta_t}$. When the value of $\frac{1}{\beta_t}$ becomes less than 1, then for that iteration the weight of the boosting classification may be less than zero. Hence, we empirically fixed the iterations of AdaBoost.M1 for the four classifiers (i.e., NB, k-NB, RI and DT) to 13, 17, 11 and 9 respectively for the feature set $\{f_L, f_S, f_M\}$ since the weight of $\frac{1}{\beta_t}$ becomes less than 1 after those values. Similarly, for feature set $\{f_L, f_S\}$ and $\{f_L\}$ the iterations were 14, 19, 13, 10 and 15, 20, 15, 12 respectively for the four base classifiers. Figure 2 depicts the iteration sizes of the four classifiers in the boosting approach.

Stacking

As discussed in Section 4.3.8, in stacking one classifier plays the role of ML while the remaining classifiers act as BLs. Therefore, with four classifiers four experiments were conducted separately. The obtained accuracies are reported in Table 4.7. From the experimental results it was observed that the model trained with DT as the model learner and NB, k-NB, RI as the base learners achieved the best classification accuracy.

Table 4.7: Accuracies obtain on the fine-grained classification with stacking

Base Learner	Model Learner	Class	f_L	f_L+f_S	f_L+f_S+f_M
k-NB,RI,DT	NB	F_{PER}	79.81	81.67	82.86
		F_{ORG}	81.79	83.02	84.02
		F_{LOC}	81.97	83.74	84.91
		F_{TEM}	81.45	82.81	83.73
		F_{NUM}	81.83	82.07	83.54
		F_{METH}	82.15	83.13	84.09
		F_{REA}	82.24	83.36	84.42
		F_{DEF}	81.76	83.05	84.23
		F_{MISC}	80.21	82.33	83.21
NB,RI,DT	k-NB	F_{PER}	79.93	81.79	83.03
		F_{ORG}	81.86	83.16	84.13
		F_{LOC}	82.08	83.82	85.06
		F_{TEM}	81.52	83.01	83.87
		F_{NUM}	81.97	82.18	83.71
		F_{METH}	82.28	83.20	84.18
		F_{REA}	82.31	83.43	84.45
		F_{DEF}	81.82	83.21	84.31
		F_{MISC}	80.29	82.42	83.35
NB,k-NB,DT	RI	F_{PER}	80.56	83.06	84.22
		F_{ORG}	82.86	83.98	85.03
		F_{LOC}	80.23	81.49	82.95
		F_{TEM}	83.21	84.78	85.97
		F_{NUM}	82.37	83.42	84.77
		F_{METH}	83.54	84.93	86.27
		F_{REA}	84.03	85.75	86.73
		F_{DEF}	80.01	82.33	84.21
		F_{MISC}	82.45	83.86	84.87
NB,k-NB,RI	DT	F_{PER}	84.97	86.69	88.71
		F_{ORG}	83.32	85.06	87.43
		F_{LOC}	82.93	84.21	85.71
		F_{TEM}	84.84	86.13	87.95
		F_{NUM}	83.57	85.17	87.49
		F_{METH}	84.85	86.91	88.56
		F_{REA}	84.69	86.78	88.29
		F_{DEF}	84.38	85.65	87.51
		F_{MISC}	84.02	86.11	88.42

Voting

Unlike the ensemble approach, in the voting approach all the classifiers were applied at the same time to predict the question class. Table 4.8 tabulates the accuracies obtained with this approach.

Fine-Grained Classification Observations

Initially, we applied NB, k-NB, RI and DT classifiers separately. Then, we applied state-of-the-art classifier combination techniques on the lexical, syntactic and semantic feature sets. Table 4.6 shows that the boosting approach achieves better performance than bagging. In the stacking approaches, the setup with NB, k-NB, RI as BLs and DT as ML outperforms other setup combinations. The stacking ap-

Table 4.8: Accuracies obtain on the fine-grained classification with voting

Base Learner	Class	f_L	f_L+f_S	f_L+f_S+f_M
NB, k-NB,RI,DT	F_{PER}	79.81	81.67	82.86
	F_{ORG}	81.79	83.02	84.02
	F_{LOC}	81.97	83.74	84.91
	F_{TEM}	81.45	82.81	83.73
	F_{NUM}	81.83	82.07	83.54
	F_{METH}	82.15	83.13	84.09
	F_{REA}	82.24	83.36	84.42
	F_{DEF}	81.76	83.05	84.23
	F_{MISC}	80.21	82.33	83.21

proach outperforms the voting approach with slight margin. However, the boosting approach with base classifier DT achieves the best. It was noticed from the fine-grained question classification that all the classifier combination approaches beat the individual classifier approaches with a notable margin.

4.3.11 Discussion

Although QA research in other languages (such as English) has progressed significantly, for majority of Indian languages it is at the early stage of development. In this study, we addressed the QC task for Bengali, one of the most spoken languages in the world and the second most spoken language in India. We reported experiments for coarse-grained and fine-grained question classification. We employed lexical, syntactic and semantic features. We applied classifiers individually as well as combination approaches. The automated Bengali question classification system obtains up to 91.65% accuracy for coarse-grained classes and 87.79% for fine-grained classes using classifier combination approaches based on four classifiers, namely NB, k-NB, RI and DT. The contributions of this work are:

- 69 fine-grained classes have been introduced to the question classification taxonomy.
- This work successfully deploys state-of-the-art classifier combination approaches for the question classification task in Bengali.
- Coarse-grained classification accuracy is increased by 4.02%.
- 87.79% classification accuracy has been achieved for fine-grained classes.
- This work improves QC accuracy which in turns enhances the Bengali QA system performance.

In coarse-grained question classification, overall the voting approach with majority voting technique performs best among the four classifier combination approaches,

namely bagging, boosting, stacking, and voting. However, the stacking approach produces the best results for fine-grained classification.

The only available QA dataset for Bengali contains only 1,100 questions. In future, we would like to contribute to enlarge the dataset. One of the future directions of this study is employing the state-of-the-art neural network techniques. Also, we would like to apply the approaches used in this study to other less investigated languages.

Named Entity Extraction

Mollá et al. (2007) argued that current text-based question answering (QA) systems usually contain a named entity recognizer (NER) as a core component. The rationale of incorporating an NER as a module in a QA system is that many fact-based answers to questions are entities that can be detected by an NER. Therefore, by incorporating an NER in the QA system, the task of finding some of the answers is considerably simplified. The study of Noguera et al. (2005) also confirmed the positive impact of NE recognition in QA. Generally, the NER system is used in a QA system as a stand-alone system and the NER system is developed independently of the QA task.

Named entities (NEs) have a special status in Natural Language Processing (NLP) because of their distinctive nature which other elements of human languages do not have, e.g. NEs refer to specific things or concepts in the world and are not listed in the grammars or the lexicons. Due to the significant presence of NE in the text documents, automatic identification and classification of NEs benefit text processing. NER is a task that seeks to locate and classify NEs in a text into predefined categories such as the of persons, organizations, locations, expressions of times, quantities, etc. The NER task can be viewed as a two stage process: a) Identification of entity boundaries, b) Classification into the correct category. For example, if *'Narenda Modi'* is a named entity in the corpus, it is essential to identify the beginning and the end of this entity in the sentence. Subsequently, the entity must be classified into the predefined category, which is PERSON (Named Entity Person) in this case.

NER is of utmost importance in many NLP applications such as Machine Translation, Question-Answering, Automatic Summarization, Information Extraction, etc. The task of building an NER for Indian languages (ILs) presents various challenges

related to their linguistic characteristics. The challenges include no capitalization, unavailability of large gazetteer, relatively free word order, spelling variation, rich inflection, ambiguity, etc.

Bandyopadhyay (2008) reported that the computational research aiming at automatically identifying NEs in texts forms a vast and heterogeneous pool of strategies, techniques and representations from hand-crafted rules towards machine learning approaches. Generally, NER systems are based on one of the following approaches:

- Linguistic approaches
- Machine Learning(ML) based approaches
- Hybrid approaches

The linguistic approaches based NER systems (Grishman, 1995; McDonald, 1993; Wakao et al., 1996) typically use hand-crafted grammatical rules written by linguistics. The main disadvantages of these rule-based techniques are that they require huge experience and grammatical knowledge of the particular language or domain and these systems are not transferable to other languages or domains. On the other hand, ML based NER systems use learning algorithms that require large annotated datasets for training and testing (Hewavitharana and Vogel, 2013). An advantage of the ML based NER systems is that they are adaptable and updatable with minimal time and effort as long as sufficiently large datasets are available (Shaalan, 2014). ML methods such as Hidden Markov Model (HMM) (Bikel et al., 1997), Conditional Random Field (CRF) (Li and McCallum, 2003), Support Vector Machine (SVM) (Yamada et al., 2002), Maximum Entropy (ME) (Borthwick and Grishman, 1999) are the most widely used approaches. Besides the two above mentioned approaches, Hybrid approaches based NER systems (Saha et al., 2008) combine the strongest points from both Rule based and statistical methods.

Mainly, ML and hybrid approaches proved successful in NER for Bengali language. The survey by (Sharma et al., 2011) on NER for ILs details the various approaches used for Bengali NER by researchers. The *ML-based* work are: (Ekbal et al., 2008),(Ekbal and Saha, 2010a),(Ekbal and Saha, 2010b), (Ekbal and Saha, 2010c),(Hasanuzzaman et al., 2009),(Ekbal and Bandyopadhyay, 2008),(Ekbal and Bandyopadhyay, 2009). Whereas, Chaudhuri and Bhattacharya (2008); Gali et al. (2008); Saha et al. (2008) followed the *Hybrid-based* approach.

Although the use of MIRA in NER was noted for English (Ganchev et al., 2007), MIRA approach has not been employed in NER for any Indian languages till date. We employed MIRA for developing NER for Bengali. We identified suitable language independent and dependent features for the Bengali NER system.

5.1 Margin Infused Relaxed Algorithm

Crammer and Singer (2003) reported *Margin Infused Relaxed Algorithm*, a machine learning algorithm for multiclass classification problems. It is designed to learn a set of parameters (vector or matrix) by processing all the given training examples one-by-one and updating the parameters according to each training example, so that the current training example is classified correctly with a margin against incorrect classifications at least as large as their loss. The change of the parameters is kept as small as possible. MIRA, also called passive-aggressive algorithm (PA-I), is an extension of the perceptron algorithm for online machine learning that ensures that each update of the model parameters yields at least a margin of one. The flow of the MIRA is depicted in Figure 5.1.
A two-class version called binary MIRA simplifies the algorithm by not requiring the solution of a quadratic programming problem. When used in an one-vs-all configuration, binary MIRA can be extended to a multiclass learner that approximates full MIRA, but may be faster to train.

Initialize: Set $\mathbf{M} = 0$ $(\mathbf{M} \in \mathbb{R}^{k \times n})$.
Loop: For $t = 1, 2, \ldots, T$

- Get a new instance $\bar{x}^t \in \mathbb{R}^n$.
- Predict $\hat{y}^t = \arg\max_{r=1}^{k} \{\bar{M}_r \cdot \bar{x}^t\}$.
- Get a new label y^t.
- Set $E = \{r \neq y^t : \bar{M}_r \cdot \bar{x}^t \geq \bar{M}_{y^t} \cdot \bar{x}^t\}$.
- If $E \neq \emptyset$ update $\mathbf{M}$ by choosing any $\tau_1^t, \ldots, \tau_k^t$ that satisfy:
  1. $\tau_r^t \leq 0$ for $r \neq y^t$ and $\tau_{y^t}^t \leq 1$.
  2. $\sum_{r=1}^{k} \tau_r^t = 0$.
  3. $\tau_r^t = 0$ for $r \notin E \cup \{y^t\}$.
  4. $\tau_{y^t}^t = 1$.
- For $r = 1, 2, \ldots, k$ update: $\bar{M}_r \leftarrow \bar{M}_r + \tau_r^t \bar{x}^t$.

Output : $H(\bar{x}) = \arg\max_r \{\bar{M}_r \cdot \bar{x}\}$.

Figure 5.1: Algorithm (Crammer and Singer, 2003)

Suppose sequence $(\bar{x}^1, y^1), ..., (\bar{x}^t, y^t), ...$ is the instance-label pairs. Each instance $\bar{x}^t$ is in $\mathbb{R}^n$ and each label belongs to a finite set Y of size k. It can be assumed without loss of generality that $\mathtt{Y} = \{1, 2, ..., k\}$. A multiclass classifier is a function $H(\bar{x})$ that maps instances from $\mathbb{R}^n$ into one of the possible labels in $\mathtt{Y}$. The classifier is in the form $H(\bar{x}) = \text{argmax}_{r=1}^{k} \{\bar{M}_r . \bar{x}\}$, where $\mathbf{M}$ is a $k \times n$ matrix over the reals and $\bar{M}_r \in \mathbb{R}^n$ denotes the r^{th} row of $\mathbf{M}$. The inner product of $\bar{M}_r$ with the instance $\bar{x}$ is called the *similarity-score* for class r. Thus, the considered classifiers set the label of an instance to be the index of the row of $\mathbf{M}$ which achieves the highest *similarity-score*.

On round t the learning algorithm gets an instance $\bar{x}^t$. Given $\bar{x}^t$, the learning algorithm outputs a prediction, $\hat{y} = argmax_{r=1}^{k}\{\bar{M}_r.\bar{x}^t\}$. It then receives the correct label y^t and updates its classification rule by modifying the matrix **M**. It can be said that the algorithm made a (multiclass) prediction error if $\hat{y}^t \neq \bar{y}^t$. The goal is to make as few prediction errors as possible.
We used *miralium*[1] which is the open source java implementation of MIRA.

5.2 Features

The features, used in this NER task, can be categorized into two: language dependent and language independent features. This section outlines the features employed in this NER task.

5.2.1 Language Independent Features

As the name suggests the language independent features can be applied to any language including other Indian languages, e.g., Bengali, Hindi, Tamil, Punjabi, etc. The following language independent features were applied to the NER work.
Window of words: Previous or next words of the target word might be used to determine its category. The preceding m words and following n words along with the target word are considered to build the window. However, majority of research works used $m = n$. For this NER task, following a few trials we found that a suitable window size is five with $m = 2$ and $n = 2$.
Word Suffix: For highly inflectional languages (i.e. ILs), target word suffix information is very helpful to identify NEs. Although a stemmer or morphological analyzer can recognize the suffixs properly, fixed length suffixes can be used as features in absence of such tools. We used four fixed length suffixes of length 5, 4, 3 and 2.
Word prefix: Similar to the suffix feature, target word prefixes can also be used. We used four fixed length prefixes of length 5, 4, 3 and 2.
First word: Usually, in most of the languages the first word is the subject (or part of the subject) of the sentence. Therefore, the first word of a sentence can be used as a feature.
Word length: It has been observed that short words are rarely NEs. Therefore, length of the word may be used as a feature.
Part of Speech (POS): The POS of the target word and surrounding words may be useful feature for NER. Since NEs are noun phrases, the noun tag is very relevant.
Presence of Digit: Usually, digit combined with symbols make NEs, e.g., 12/10/2014, 55.44%, 22/-, etc. Therefore, presence of digit in the target word is

[1]https://code.google.com/p/miralium/

a very useful feature. This feature could be used to identify time expression, measurement and numerical quantities.

5.2.2 Language Dependent Features

Language dependent features increase the accuracy of NER systems. Therefore, in most of the experiments they are used along with language independent features. We used the following language dependent features.

Clue words: Usually, clue words occur before or after the NEs. These words play a useful role to identify NEs. For example, *'Mr.'* is most likely present before starting a person name. Similarly, *'Limited'* (or '*Ltd.*') is most likely present after an organization. Generally, a list of clue words is prepared for NER task and list-lookup technique is employed. In this NER task, we also prepared two clue word lists, namely person clue list and organization clue list. The lists were compiled under human supervision from the archive (100 documents) of an online available widely used Bengali newspaper. Person and organization clue word lists contain 39 and 53 words, respectively. We used this feature as a binary feature. If the target word is present in (any of) the lists, then the value is set to 1, otherwise 0.

Gazetteers list: The use of gazetteers list is very common in NER task. We manually prepared four lists, namely names of months, names of sessions, Days of a week, and names of units.

5.3 Experiments

In this sub-section, we discuss our study of NER for Bengali using language independent and dependent features employing MIRA. Also, we discuss the comparative study with the existing Bengali NER models. In this task, we considered the *name finder* tool reported in Singh (2008a) as our baseline; *name finder* is open source, Maximum Entropy based and part of the OpenNLP[2] package. Initially, we carried out the experiments with language independent features only. Next, we used language independent and dependent features together. We noted that the use of language dependent features increases the overall F-measure (by 3.12%).

5.3.1 Corpus and Tagset

In this NER task, we used the dataset released in the NER shared task organized in the *IJCNLP-08 Workshop on South and South East Asian Languages (NERSSEAL)*[3]. The dataset contains over six thousand sentences as training examples and almost two thousand sentences as test sentences. A total of 5000 NEs are present in the training data. Corpus statistics are shown in Table .

[2]http://opennlp.sourceforge.net/

[3]http://ltrc.iiit.ac.in/ner-ssea-08/index.cgi?topic=5

	Training	Testing
Sentences	6030	1835
Words	112845	38708
NEs	5000	1723

Table 5.1: NERSSEAL corpus statistics.

We also employed the *tagset*[4] which was used in NERSSEAL. The tagset consists of 12 NE tags. Tagset and tag-wise statistics are given in Table 5.2 .

Tag	Name	Example	Training	Testing
NEP	Person	Bob Dylan	1,299	728
NED	Designation	President, Chairman	185	11
NEO	Organization	State Government	264	20
NEA	Abbreviation	NLP, I.B.M.	111	9
NEB	Brand	Pepsi, Windows	22	0
NETP	Title-Person	Mahatma, Dr., Mr.	68	57
NETO	Title-Object	American Beauty	204	46
NEL	Location	New Delhi, Paris	634	202
NETI	Time	10th July, 5 pm	285	46
NEN	Number	3.14, Fifty five	407	144
NEM	Measure	three days , 5 kg	352	146
NETE	Terms	Horticulture	1,165	314

Table 5.2: NERSSEAL NE tagset and statistics.

5.4 Results

We evaluated the performance of the system in terms of the standard precision, recall, and F-Measure as follows:

Precision: $P = \frac{c}{r}$

Recall: $R = \frac{c}{t}$

F-Measure: $F_{\beta=1} = \frac{2 \times P \times R}{P+R}$

where c is the number of correctly retrieved (identified) NEs, r is the total number of NEs retrieved by the system (correct plus incorrect) and t is the total number of NEs in the test data. Initially using the language independent features, we obtained 89.26%, 82.99% and 86.01% precision, recall and F-measure respectively. Employing language independent and dependent features, we obtained 91.20%, 87.17% and 89.13% precision, recall and F-measure respectively. NE tag specific results are shown in Table 5.3. It is evident from the Table that the use of language dependent

[4]http://ltrc.iiit.ac.in/ner-ssea-08/index.cgi?topic=3

features enhances the performance of the NER model for all the NE classes. The NER model idenfied NEP class with the highest precision of 92.82% among the NE classes. In the NERSSEAL testset, no samples of NEB class were present. Therefore, we could not evaluate the NEB class.

	Lang. Independent			Lang. Dependent		
Tag	P	R	$F_{\beta=1}$	P	R	$F_{\beta=1}$
NEP	92.86	89.29	91.04	94.20	91.48	92.82
NED	66.67	36.36	47.06	70.00	63.64	66.67
NEO	73.68	70.00	71.79	78.95	75.00	76.92
NEA	37.50	33.33	35.29	42.86	33.33	37.50
NEB	NP	NP	NP	NP	NP	NP
NETP	82.35	73.68	77.78	83.93	82.46	83.19
NETO	80.95	73.91	77.27	84.09	80.43	82.22
NEL	89.67	81.68	85.49	91.94	84.65	88.14
NETI	84.21	69.57	76.19	88.37	82.61	85.39
NEN	88.89	77.78	82.96	92.86	81.25	86.67
NEM	88.28	77.40	82.48	90.85	88.36	89.58
NETE	87.00	83.12	85.02	88.60	86.62	87.60

Table 5.3: Evaluation results on different NE classes. (NP: Not present in reference data)

5.5 Comparisons with Existing Systems

In Table 5.4, we have reported the existing Bengali NER systems which used the same corpus and evaluation metrics as adopted in this work; i.e., NERSSEAL shared task data and evaluation metrics. It is evidient from Table 5.4 that the proposed NER system outperforms the existing models which are based on various machine learning approaches. The best performing existing NER system in Bengali is based on SVM. Our proposed NER system for Bengali outperformed the existing NER system by 4.98%. The reasons behind the superior performance of the proposed MIRA based NER system are the better optimization technique of MIRA and its ability to handle the overlapping features efficiently than the existing systems. Basically, MIRA does not explicitly optimize any function. Therefore, there is no involvement of probabilistic interpretation.

5.6 Discussion

The proposed system for Bengali NER is based on MIRA which uses both language-independent and language-dependent features. The obtained results show that the proposed system outperforms the existing systems based on other machine learning

Model	F-Measure
Baseline (Singh (2008a))	12.30%
Praveen and Ravi (2008)	39.77%
Gali et al. (2008)	40.63%
Ekbal and Bandyopadhyay (2008)	59.39%
Saha et al. (2008)	65.95%
Ekbal and Bandyopadhyay (2010)	84.15%
MIRA	**89.13%**

Table 5.4: Comparative evaluation results.

approaches. However, this system was unable to identify NEA (i.e., Abbreviation) properly due to our assumption that short words are rarely NEs which is not true in this case. One of the possible soultions could be post-processing with heuristic patterns. Before this task, MIRA has been applied to neither Bengali nor any other ILs. Therefore, we not only improve the accuracy but also successfully apply MIRA in the NER task for one of the ILs. Incorporating MIRA in Bengali can be considered as one of the notable contributions of this work. MIRA may be used for other ILs to enhance the performance of state-of-the-art NER systems.

The performance of the proposed NER system may be enhanced further by applying post-processing with a set of heuristics and Ensemble approaches. Also one of the extension of MIRA, e.g., AdaGrad, may be used to improve performance further for NER systems.

6

BFQA System

A QA system is an automatic system capable of answering natural language questions in a human-like manner: with a concise, precise answer. Designing a QA system for European languages particularly for English is not new in natural language processing. However, there exists no Bengali QA system till date and the development of Bengali QA system is at its nascent stage. Generally, questions can be classified into five broad categories (Ittycheriah et al., 2000; Zheng, 2002a): factoid questions, list questions, definition questions, complex questions and speculative questions. We have proposed a prtotype for Bengali QA system which can answer factoid questions.

6.1 BFQA Architecture

We have proposed a factoid QA system for Bengali. We refer to this system as Bengali Factoid Question Answering (BFQA) system. BFQA system has a pipeline architecture having three components: question analysis, sentence extraction and answer extraction. The first component (i.e., question analysis module) accepts natural language question in Bengali posed by the user as input. After receiving the natural language question, the question analysis module processes it in five stages: question type (QType) identification, expected answer type (EAT) identification, named entity identification, question topical target (QTT) identification and keyword identification. Then, the valid keywords are 'AND'ed together to form the query. Based on the query, sentences are extracted from the passages. Next, the extracted sentences are ranked based on the answer score value. Finally, the extracted answers are validated using the EAT module. The architecture of the proposed model is depicted in Figure 6.1.

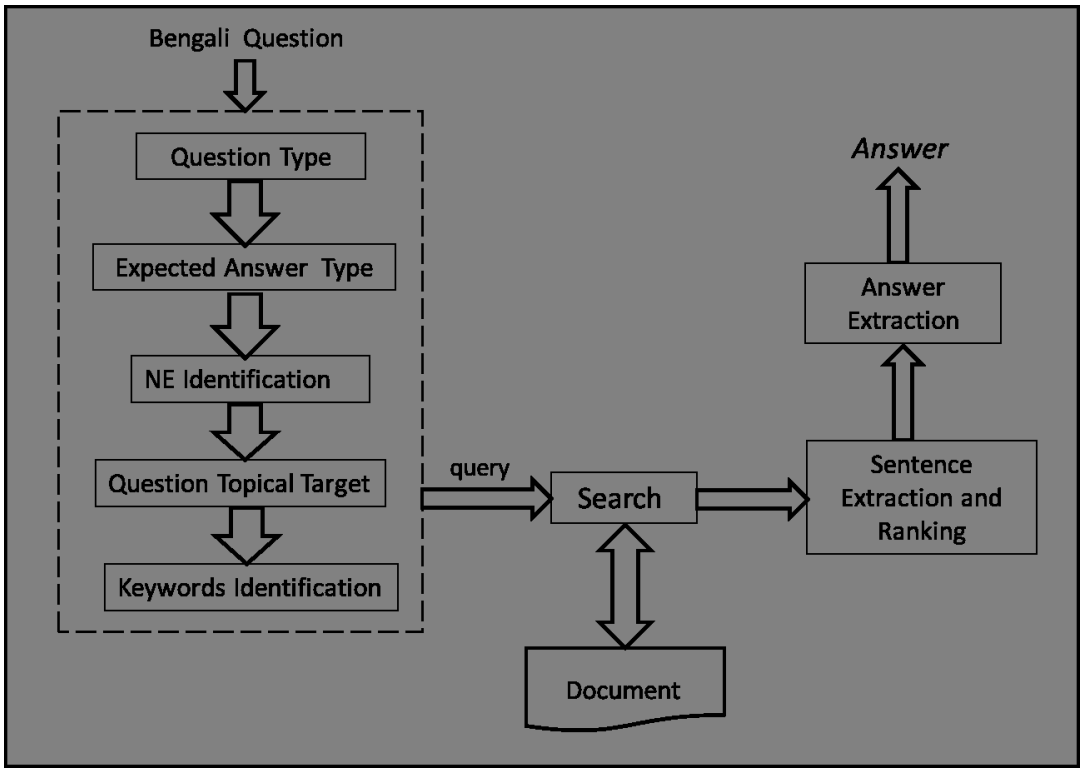

Figure 6.1: BFQA architecture.

6.2 Question Analysis

Question analysis plays a crucial role in an automatic QA system. Acquiring the information embedded in a question is a primary task that allows the QA system to decide the appropriate strategy in order to provide the correct answer to the question. Chen et al. (2000) stated that when the question analysis module fails, it is hard or almost impossible for a QA system to provide the correct answer. In this work, the question analysis task is divided into five parts: question type identification, inferring the expected answer type, obtaining the question topical target, named entity (NE) extraction and identification of keywords.

Question Type Identification: Question Type (QType) identification or question classification is a crucial component of every QA System. Depending on the classification strategy, the task of a question classifier is to assign one or more predefined class labels to a given question written in natural language. The set of predefined categories which are considered as question classes is usually referred to as the question taxonomy or answer type taxonomy. For this work, we used a single layer taxonomy which contains nine coarse-grained classes. The taxonomy, that was proposed in our work (Banerjee and Bandyopadhyay, 2012a), is the only standard taxonomy available in Bengali QA research. For QType identification, we employed the approach described in Banerjee and Bandyopadhyay (2013) which is the best reported work (91.65% accuracy) yet for Bengali QType identification task.

Expected Answer Type Identification: Prager et al. (2007) defined the Expected Answer Type (EAT) as the class of object or rhetorical type of sentence required by the question. EAT can also be defined as the semantic category associated with the desired answer, chosen from a predefined set of labels. A number of QA systems (Chen et al., 2000; Hovy et al., 2000a; Moll, 2003) successfully used a set of hand-crafted rules for finding the EAT. The rules were written using the regular expressions (RE). Every RE is associated with an EAT that is to be assigned to a question if it matches its pattern. The disadvantage of hand-crafted rules is that those are very specific to the particular domain. However, these rules are very useful at the initial stage of the development. In this work, we also used hand-written rules to determine the EAT of a question.

Named Entity Identification: NEs play a crucial role in QA since the answer to a factoid question is an NE. Moreover, NEs present in the question most likely are present in the sentence that contains the expected answer. Therefore, NE identification in the question is a very essential task. However, unfortunately any QA specific NER system is not available for Bengali. We employed the NER system described in 5) which reported 65.95% accuracy.

Question Topical Target Identification: Moldovan et al. (2000) argued that knowing the QType alone is not sufficient for finding answers to questions. Question Topical Target (QTT) (sometimes also referred to as question focus (Monz, 2003), or question topic (Prager et al., 2007)) corresponds to a noun or a noun phrase that is likely to be present in the answer. Proper identification of QTT benefits the question answering process, since QTT terms or their synonyms are likely to appear in a retrieved sentence that contains the answer. Due to varying position of Bengali interrogatives, it is very challenging to separate the QTT when the interrogative appears within the noun phrase. To follow the same strategy for all cases, Bengali QTT is defined as a comma separated named entity list in this work.

Keywords Identification: The keywords of the posed natural language question are used to form a query which is used to extract passages that might contain the expected answer. In the absence of any full-fledged parser, a *shallow parser* for Bengali is used to parse the Bengali questions. We considered the nouns, proper nouns (i.e., NEs), verbs and adjectives as legitimate keywords. We considered the other words as stop words. In QTT identification stage, we included the NEs. Therefore, in this step, we excluded the NEs. Rather, we formed the query by merging QTT with nouns, verbs and adverbs present in the question. Successively, the valid keywords are 'AND'ed together to form the query.

6.3 Sentence Extraction and Ranking

In this work, we used the corpus (Banerjee et al., 2014a) which has the constraint that a question is related to a particular document only. Therefore, we also imposed two constraints on the searching technique: i) the answer to the posed question is searched from a single document, and ii) each of the paragraphs of the document is treated as individual documents while searching. The objective of this module is to extract the relevant sentences that contain the expected answer from different paragraphs of a single document. The relevant sentences are searched using the query. Also, the relevant sentences are searched taking into account the imposed constraints.

After extracting the sentences which contain the expected answer, these are passed to the NER system (Banerjee et al., 2014b) which we developed particularly for Bengali QA system. The NER system tags the NEs present in the question text. Based on the *answer score*, the extracted sentences are ranked. The *answer score* is calculated using the EAT and *similarity score*. The *answer score* is calculated based on: (i) *syntactic similarity*, (ii) *name proportions*, and (iii) *paragraph relevancy*.

Syntactic Similarity: if Q_t is the natural language (NL) question and is composed of n words, then the question text Q_t can be expressed as:

$Q_t = Q_1 Q_2 Q_3 Q_4 ... Q_{n-2} Q_{n-1} Q_n$

Let $V_Q = Q_1, Q_2, Q_4, Q_7, Q_8, ... Q_n$, and $V_{Stop} = Q_3, Q_5, Q_6, Q_9, ...$;

Then $Q_t = V_Q \cup V_{Stop}$ and $V_Q \cap V_{Stop} = \emptyset$; where V_Q and V_{Stop} are the two word vectors, namely content and stop words respectively.

V_S is the vector which contains all the sentences in the document of p sentences , i.e. $V_S = V_{S_1}, V_{S_2}, V_{S_3}, V_{S_4}, ... V_{S_{P-1}}, V_{S_P}$; where V_{S_k} represents the k^{th} sentence in the document. Let V_{S_k} contain c words.

$V_{S_k} = w_1, w_2, w_3, w_4, ..., w_{c-1}, w_c$

In *similarity measure*, we only consider the part-of-speech (POS): verb (VB), noun (NN), adjective (ADJ) and proper noun (NE). Let {VB, NN, ADJ, NE ∈ POS}. Four weights λ_{vb}, λ_{nn}, λ_{adj} and λ_{ne} have been defined corresponding to the verb, noun, adjective and named entity, respectively. We have set $\lambda_{vb} = 0.2$, $\lambda_{np} = 0.3$, $\lambda_{adj} = 0.1$ and $\lambda_{ne} = 0.4$ (so that the four weights add up to 1), i.e., $\sum_{Pos \in \{vb,np,adj,ne\}} \lambda_{Pos} = 1$.

Therefore, the similarity of the NL question Q_t and a sentence S_l is calculated by the following formula: $Similarity_{(Q_t,S_l)} = \sum_{K=1}^{n} frequency_Q . w_K$;

where, $w_K = Q(\lambda_{Pos})$ and $frequency_Q$ is the number of occurrence of question word Q in the sentence S_l.

Name proportions (nprop): We used *Jaccard similarity coefficient* to measure *name proportion*. *Jaccard similarity coefficient* is a set similarity measure that compares the similarity between two feature sets. In *name proportion* measure, it

is defined as the size of the intersection of the named entities in the question and a sentence normalized by the size of the union of the named entities in the question and the sentence.

Paragraph relevancy: We measured the relevancy of a paragraph to a question by counting the presence of query words in that paragraph. We also considered the synonyms of the query words for measuring paragraph relevancy. A *relevancy weight* is assigned to each appearance to distinguish between original query word and synonymous words .

$$\textit{relevancy weight}(r_w) = \begin{cases} 1.0 & \text{if original query term appears in the paragraph} \\ 0.9 & \text{if synonym appears in the paragraph} \\ 0.0 & \text{neither query word nor any synonym} \end{cases}$$

Each of the words in the paragraph is considered for paragraph relevancy calculation. Paragraph relevancy of a word is calculated as follows.

$$\begin{aligned} R_w &= \textit{frequency} \times \textit{relevancy weight} \\ &= f_w \times r_w \end{aligned}$$

Therefore, if a paragraph contains k distinct words, then the *paragraph relevancy* for that paragraph can be measured using the following formula.

$$\begin{aligned} R_p &= f_{w_1} \times r_{w_1} + f_{w_2} \times r_{w_2} + f_{w_3} \times r_{w_3} + \ldots + f_{w_k} \times r_{w_k} \\ &= \sum_{i=1}^{k} f_{w_i} \times r_{w_i} \\ &= \text{sum of the paragraph relevancy for each distinct word in the paragraph} \end{aligned}$$

Finally, the score for the three metrics are summed up to arrive at the answer score.

answer score = syntactic similarity + name proportion + paragraph relevancy

6.4 Answer Extraction

The last module of the QA pipeline architecture is the Answer Extraction module. Using the earlier module, the sentences are ranked based on *answer score*. The answer to the natural language question is determined by the NE which is suggested by EAT in the question analysis module. Here, three cases are possible:

- **Case-1:** Only a single NE in the sentence is of suggested NE type by EAT.
- **Case-2:** Multiple NEs having the same NE tag suggested by EAT.
- **Case-3:** No NE in the sentence having NE tag suggested by EAT.

In the first case, the answer extraction is trivial and the NE word suggested by EAT is the answer to the question. However, the second case is a bit ambiguous and we

Type	Geography	Agriculture	Overall
Person	22	0	22
Organization	4	6	10
Location	6	11	17
Temporal	47	13	60
Numerical	60	15	75

Table 6.1: QType statistics.

need to apply some extra effort. We use a novelty factor to solve this ambiguity, i.e., choose the NE as candidate answer which is not present in the query. In the third case, the QA system simply fails to answer the question.

6.5 Experiments

We developed the QA system which is focused on Bengali factoid questions. During this QA system development, our corpus development for Bengali QA research was in process. Moreover, no QA corpus for Bengali was available. Therefore, we compiled a small corpus for Bengali QA research. A brief description of the used corpus is given below.

Corpus: We compiled fourteen documents in the geography and agriculture domains which were acquired from the Wikipedia. In-house 3 Bengali language experts were involved in this small corpus development work. A total of 184 factoid questions were prepared and annotated according to three question answering based levels, namely Question Class (kappa - 0.91), Expected Answer Type (kappa - 0.85) and Question Topical Target (kappa - 0.89). Corpus statistics are given in Table 6.1.

Results: There are many evaluation metrics such as Recall and Precision, Mean Reciprocal Rank (MRR), Confidence Weighted Score, K-Measure, etc., which are used for evaluating QA systems. The MRR evaluation metric, used for factoid QA, was introduced in the TREC QA track in 1999. We chose MRR metric to evaluate the performance of BFQA system. MRR is formulated as follows:

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

where, $rank_i$ represents the rank of the correct answer for the i^{th} question and $|Q|$ is the number of test questions. The evaluation results for our BFQA system are shown in Table 6.2.

6.6 Discussion

The BFQA system was the first attempt to build a factoid QA system for Bengali. We proposed an architecture to address the scenarios common to low-resource lan-

Domain	**#Documents**	**#Questions**	**MRR**
Geography	10	139	0.34
Agriculture	4	45	0.31
Overall	14	184	0.32

Table 6.2: Corpus statistics and system evaluation

guages particularly for Indian languages. Also, we discussed the major challenges of developing a QA system for Bengali. A sentence ranking strategy was also proposed for the Bengali QA system. It can be observed from Table 6.2 that the accuracy of the BFQA system is not at par with those for the European languages. The probable reasons for the somewhat poor performance of the system can be attributed to the low accuracies of the shallow parser and the NER system as the accuracy of factoid QA system is largely dependent on the performance of the NER component and the parser. In absence of any gold standard test set in Bengali, we also prepared our own test set to evaluate the system.

Conclusion

This chapter concludes this thesis by revisiting the different findings and proposing the future line of research. As discussed in the literature review (cf. Chapter 2), in spite of the significant development in the field of QA, any notable progress have not been seen in QA research for Indian languages. Unfortunately, there is no QA system developed for Bengali language. However, the need of QA system is realized on various domains.

The various efforts in the field of NLP in Bengali have resulted in the development of many resources and tools, especially for handling basic tasks such as POS-tagging, morphological analysis, phrase chunking, syntactic parsing, etc. The advancements in Bengali NLP call for the development of next generation search engine, i.e., QA system. On the other hand, we have seen in the Chapter 2 that the 21^{st} century has witnessed the emergence of QA devoted to other languages. In the framework of the evaluation campaigns like TREC, CLEF etc., various approaches were proposed and tested for different languages particularly for English. One of the simplest approaches is surface-based approach which extracts text similar to the question in terms of keywords and(or) structure. Besides this simple approach, the evaluation campaigns were held regularly for new approaches and QA research noticed a trend towards the introduction of new approaches based on semantic comparison rather than surface comparison. The prime motivation behind the new approaches was to develop QA systems that can answer those questions other than the factoid ones. However, in this work, we have developed a QA system that can address the factoid questions which are posed in Bengali.

The research questions regarding the monolingual Bengali QA are identified in Section 1.3 of Chapter 1. The subsequent chapters successfully addresses the identified research questions.

In this thesis, we have presented our approach towards building an efficient Bengali QA system with the ability of dealing with the specific challenges related to:

- acquisition as well as annotation of the Bengali corpus for QA research,
- processing of the Bengali questions,
- extracting answers from Bengali document collections,
- the users' expectations to automatically answer factoid questions, and
- evaluation of the system.

It is expected that the approaches employed in this work will advance the field of QA and inspire the QA community to develop QA systems that can be used by the general public to access the growing source of knowledge available as free text. Furthermore, it is also expected that the thoughts presented in this thesis will motivate the QA researchers to propose novel techniques for QA that will be even more effective than the approaches employed in this thesis. Thus, it will ultimately lead to the creation of more efficient and accurate QA systems.
The research work which is carried out in this thesis leads to these following contributions.

- We contributed to the Bengali QA research by developing a dataset with an annotation scheme. The statistics of the dataset is given in Table 3.2. This dataset is not only regarded as the first corpus but also it is the sole corpus in Bengali till date. A total of 2,257 factoid questions were annotated from three domains, namely history, geography and agriculture.
- Particularly for the QA system, a NE recognition system has been developed employing Margin Infused Relaxed Algorithm (MIRA). The efficiency of the NE recognition system was evaluated as 91.23%, 87.29% and 89.69% in terms of precision, recall and F-measure, respectively.
- We have proposed a 2-layer taxonomy for Bengali QA. The taxonomy consists of 9 coarse-grained along with 62 fine-grained question classes for Bengali. Moreover, for classifying questions, we have proposed two machine learning approaches: i) individual classifier approach, and ii) classifier combination approach.
- As part of the research, a QA system framework for answering factoid Bengali question has been proposed and developed, namely BFQA. Within the BFQA framework, sentence selection and ranking scheme have also been proposed.

7.1 Future Directions

The research carried out in this thesis provide avenues for several possible directions of future work.

- The research proposed in this thesis opens up an entirely new array of research avenues for Indian QA research. The similar approach could be used to develop QA system for other Indian languages as well.
- The QA system developed in this work is restricted to handle questions of type factoid. This can be extended to handle other than factoid questions such as procedural, causal, yes-no, etc.
- One of the most obvious future work is to increase the size of the corpus. This involves incorporating more domains with factoid and non-factoid questions.
- Another line of research is to propose efficient passage retrieval and answer selection techniques which could enhance the performance of the QA system.
- Nearly for the last decade, social media has become an integral part of the human life and the use of social networking platforms such as Facebook, Twitter, WhatsApp, Instagram, etc has increased to a great extent. Due to various socio cultural reasons, instead of using native Bengali script, the native Bengali speakers use Roman script while writing in their social media interaction. This ever-increasing resource could be used as a potential resource for research for the low resourced languages such as Bengali. Therefore, in future, we would like to develop a QA system within the context of Bengali English code mixed cross script social media content.

Bibliography

Lahsen Abouenour, Karim Bouzoubaa, and Paolo Rosso. IDRAAQ: New Arabic question answering system based on query expansion and passage retrieval. CELCT, 2012.

Carlos Amaral, Helena Figueira, André Martins, Afonso Mendes, Pedro Mendes, and Cláudia Pinto. Priberam's question answering system for portuguese. In *Workshop of the Cross-Language Evaluation Forum for European Languages*, pages 410–419. Springer, 2005.

Ion Androutsopoulos, Graeme D Ritchie, and Peter Thanisch. Natural language interfaces to databases–an introduction. *Natural language engineering*, 1(1):29–81, 1995.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *In International Conference on Learning Representations (ICLR)*, 2014.

Sivaji Bandyopadhyay. Multilingual named entity recognition. In *Proceedings of the IJCNLP 2008 Workshop on NER for South and South East Asian Languages, Hyderabad, India*, 2008.

Somnath Banerjee and Sivaji Bandyopadhyay. Bengali question classification: Towards developing qa system. In *24th International Conference on Computational Linguistics*, page 25, 2012a.

Somnath Banerjee and Sivaji Bandyopadhyay. Bengali question classification: Towards developing qa system. In *Proceedings of the 3rd Workshop on South and Sotheast Asian Language Processing (SANLP), COLING*, pages 25–40, 2012b.

Somnath Banerjee and Sivaji Bandyopadhyay. Ensemble approach for fine-grained question classification in bengali. In *Proceedings of 27th Pacific Asia Conference on Language, Information, and Computation (PACLIC), Taiwan*, pages 75–84, 2013.

Somnath Banerjee, Pintu Lohar, Sudip Kumar Naskar, and Sivaji Bandyopadhyay. The first resource for bengali question answering research. In *International Conference on Natural Language Processing*, pages 290–297. Springer, 2014a.

Somnath Banerjee, Sudip Kumar Naskar, and Sivaji Bandyopadhyay. Bengali named entity recognition using margin infused relaxed algorithm. In *International Conference on Text, Speech, and Dialogue*, pages 125–132. Springer, 2014b.

Yassine Benajiba, Paolo Rosso, and Abdelouahid Lyhyaoui. Implementation of the arabiqa question answering system's components. In *Proc. Workshop on Arabic Natural Language Processing, 2nd Information Communication Technologies Int. Symposium, ICTIS-2007, Fez, Morroco, April*, pages 3–5, 2007.

Farah Benamara. Cooperative question answering in restricted domains: the webcoop experiment. In *Proceedings of the Conference on Question Answering in Restricted Domains*, 2004.

Silvia Bernardini, Marco Baroni, Stefan Evert, et al. A wacky introduction. *WaCky*, pages 9–40, 2006.

Tim Berners-Lee, Robert Cailliau, Jean-François Groff, and Bernd Pollermann. World-wide web: The information universe. *Internet Research*, 20(4):461–471, 2010.

Daniel M Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. Nymble: a high-performance learning name-finder. In *Proceedings of the fifth conference on Applied natural language processing*, pages 194–201. Association for Computational Linguistics, 1997.

MS Bindu and Idicula Sumam Mary. *Design And Development Of A Named Entity Based Question Answering System For Malayalam Language*. PhD thesis, Cochin University Of Science And Technology, 2012.

Phil Blunsom, Krystle Kocik, and James R Curran. Question classification with log-linear models. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 615–616. ACM, 2006.

Andrew Borthwick and Ralph Grishman. *A maximum entropy approach to named entity recognition*. PhD thesis, New York University, Graduate School of Arts and Science, 1999.

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996a.

Leo Breiman. Bias, variance, and arcing classifiers. 1996b.

Leo Breiman. Stacked regressions. *Machine learning*, 24(1):49–64, 1996c.

John Burger, Claire Cardie, Vinay Chaudhri, Robert Gaizauskas, Sanda Harabagiu, David Israel, Christian Jacquemin, Chin-Yew Lin, Steve Maiorano, George Miller, et al. Issues, tasks and program structures to roadmap research in question & answering (q&a). In *Document Understanding Conferences Roadmapping Documents*, pages 1–35, 2001.

Elena Cabrio, Bonaventura Coppola, Roberto Gretter, Milen Kouylekov, Bernardo Magnini, and Matteo Negri. Question answering based annotation for a corpus of spoken requests. In *Proceedings of the workshop on the Semantic Representation of Spoken Language, Salamanca, Spain, November*, 2007.

Bidyut Baran Chaudhuri and Suvankar Bhattacharya. An experiment on automatic detection of named entities in bangla. In *IJCNLP*, pages 75–82, 2008.

Jiangping Chen, Anne R Diekema, Mary D Taffet, Nancy McCracken, and Necati Ercan Ozgencil. Question answering: Cnlp at the trec-10 question answering track. 2000.

Brown Corpus. A standard corpus of present-day edited american english, for use with digital computers (brown), 1979.

Koby Crammer and Yoram Singer. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3(Jan):951–991, 2003.

Mihael Cudic, Ryan Burt, Eder Santana, and Jose C Principe. A flexible testing environment for visual question answering with performance evaluation. *Neurocomputing*, 291:128–135, 2018.

Gilles-Maurice De Schryver. Web for/as corpus: A perspective for the african languages. *Nordic Journal of African Studies*, 11(2):266–282, 2002.

Gursharan Singh Dhanjal, Sukhwinder Sharma, and Paramjot Kaur Sarao. Gravity based punjabi question answering system. *International Journal of Computer Applications*, 147(3), 2016.

Sapan Diwakar, Pulkit Goyal, and Rohit Gupta. Transliteration among indian languages using wx notation. In *Proceedings of the Conference on Natural Language Processing 2010*, number EPFL-CONF-168805, pages 147–150. Saarland University Press, 2010.

Tiansi Dong, Ulrich Furbach, Ingo Glöckner, and Björn Pelzer. A natural language question answering system as a participant in human q&a portals. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 2430, 2011.

Sanjay K Dwivedi and Vaishali Singh. Research and reviews in question answering system. *Procedia Technology*, 10:417–424, 2013.

Asif Ekbal and Sivaji Bandyopadhyay. Bengali named entity recognition using support vector machine. In *IJCNLP*, pages 51–58, 2008.

Asif Ekbal and Sivaji Bandyopadhyay. Voted ner system using appropriate unlabeled data. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, pages 202–210. Association for Computational Linguistics, 2009.

Asif Ekbal and Sivaji Bandyopadhyay. Named entity recognition using support vector machine: A language independent approach. *International Journal of Electrical, Computer, and Systems Engineering*, 4(2):155–170, 2010.

Asif Ekbal and Sriparna Saha. Weighted vote based classifier ensemble selection using genetic algorithm for named entity recognition. In *NLDB*, pages 256–267. Springer, 2010a.

Asif Ekbal and Sriparna Saha. Classifier ensemble using multiobjective optimization for named entity recognition. In *Proceedings of the 2010 conference on ECAI 2010: 19th European Conference on Artificial Intelligence*, pages 783–788. IOS Press, 2010b.

Asif Ekbal and Sriparna Saha. Maximum entropy classifier ensembling using genetic algorithm for ner in bengali. In *LREC*, 2010c.

Asif Ekbal, Rejwanul Haque, Amitava Das, Venkateswarlu Poka, and Sivaji Bandyopadhyay. Language independent named entity recognition in indian languages. In *IJCNLP*, pages 33–40, 2008.

David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. Building watson: An overview of the deepqa project. *AI magazine*, 31(3): 59–79, 2010.

William H Fletcher. Making the web more useful as a source for linguistic corpora. *Language and Computers*, 52:191–206, 2004.

William H Fletcher et al. Concordancing the web with kwicfinder. In *Third North American Symposium on Corpus Linguistics and Language Teaching*, pages 1–16. Citeseer, 2001.

Yoav Freund, Robert E Schapire, et al. Experiments with a new boosting algorithm. 1996.

Karthik Gali, Harshit Surana, Ashwini Vaidya, Praneeth Shishtla, and Dipti Misra Sharma. Aggregating machine learning and rule based heuristics for named entity recognition. In *IJCNLP*, pages 25–32, 2008.

Kuzman Ganchev, Fernando Pereira, Mark Mandel, Steven Carroll, and Peter White. Semi-automated named entity annotation. In *Proceedings of the linguistic annotation workshop*, pages 53–56. Association for Computational Linguistics, 2007.

Arnaud Grappy and Brigitte Grau. Answer type validation in question answering systems. In *Adaptivity, Personalization and Fusion of Heterogeneous Information*, pages 9–15. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE, 2010.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Bert F Green Jr, Alice K Wolf, Carol Chomsky, and Kenneth Laughery. Baseball: an automatic question-answerer. In *Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference*, pages 219–224. ACM, 1961a.

Bert F Green Jr, Alice K Wolf, Carol Chomsky, and Kenneth Laughery. Baseball: an automatic question-answerer. In *Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference*, pages 219–224. ACM, 1961b.

Ralph Grishman. The nyu system for muc-6 or where's the syntax? In *Proceedings of the 6th conference on Message understanding*, pages 167–175. Association for Computational Linguistics, 1995.

Barbara J Grosz, Karen Sparck Jones, and Bonnie Lynn Webber. Readings in natural language processing. 1986.

Sanda Harabagiu and Dan Moldovan. Question answering. In *The Oxford Handbook of Computational Linguistics*. 2003.

Sanda M Harabagiu, Steven J Maiorano, and Marius A Paşca. Open-domain textual question answering techniques. *Natural Language Engineering*, 9(3):231–267, 2003.

Mohammad Hasanuzzaman, Asif Ekbal, and Sivaji Bandyopadhyay. Maximum entropy approach for named entity recognition in bengali and hindi. *International Journal of Recent Trends in Engineering*, 1(1):408–412, 2009.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701, 2015.

Ulf Hermjakob, Eduard Hovy, and Chin-Yew Lin. Automated question answering in webclopedia: a demonstration. In *Proceedings of the second international conference on Human Language Technology Research*, pages 370–371. Morgan Kaufmann Publishers Inc., 2002.

Sanjika Hewavitharana and Stephan Vogel. Extracting parallel phrases from comparable data. In *Building and Using Comparable Corpora*, pages 191–204. Springer, 2013.

Daniel Hewlett, Alexandre Lacoste, Llion Jones, Illia Polosukhin, Andrew Fandrianto, Jay Han, Matthew Kelcey, and David Berthelot. Wikireading: A novel large-scale language understanding task over wikipedia. In *In Association for Computational Linguistics (ACL)*, pages 1535–1545, 2016.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The goldilocks principle: Reading children's books with explicit memory representations. 2015.

Lynette Hirschman and Robert Gaizauskas. Natural language question answering: the view from here. *natural language engineering*, 7(4):275–300, 2001.

Eduard H Hovy, Laurie Gerber, Ulf Hermjakob, Michael Junk, and Chin-Yew Lin. Question answering in webclopedia. In *TREC*, volume 52, pages 53–56, 2000a.

Eduard H Hovy, Laurie Gerber, Ulf Hermjakob, Michael Junk, and Chin-Yew Lin. Question answering in webclopedia. In *TREC*, volume 52, pages 53–56, 2000b.

Haiqing Hu. A study on question answering system using integrated retrieval method. *Unpublished Ph. D. Thesis, The University of Tokushima, Tokushima*, 2006.

Zhiheng Huang, Marcus Thint, and Zengchang Qin. Question classification using head words and their hypernyms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 927–936. Association for Computational Linguistics, 2008.

Zhiheng Huang, Marcus Thint, and Asli Celikyilmaz. Investigation of question classifier in question answering. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 543–550. Association for Computational Linguistics, 2009.

Nitin Indurkhya and Fred J Damerau. *Handbook of natural language processing*, volume 2. CRC Press, 2010.

Masashi Inoue and Toshiki Akagi. Collecting humorous expressions from a community-based question-answering-service corpus. In *LREC*, pages 1836–1839, 2012.

Hideki Isozaki, Katsuhito Sudoh, and Hajime Tsukada. Ntt's japanese-english cross-language question answering system. In *NTCIR*, 2005.

Abraham Ittycheriah, Martin Franz, Wei-Jing Zhu, Adwait Ratnaparkhi, and Richard J Mammone. Ibm's statistical question answering system. In *TREC*, 2000.

Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III. A neural network for factoid question answering over paragraphs. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 633–644, 2014.

Ghassan Kanaan, Awni Hammouri, Riyad Al-Shalabi, and Majdi Swalha. A new question answering system for the arabic language. *American Journal of Applied Sciences*, 6(4):797, 2009.

Boris Katz. Annotating the world wide web using natural language. In *Computer-Assisted Information Searching on Internet*, pages 136–155. Le Centre de Hautes Etudes Internationales Ðinformatique Documentaire, 1997.

Boris Katz, Sue Felshin, Deniz Yuret, Ali Ibrahim, Jimmy Lin, Gregory Marton, Alton Jerome McFarland, and Baris Temelkuran. Omnibase: Uniform access to heterogeneous data for question answering. In *International Conference on Application of Natural Language to Information Systems*, pages 230–234. Springer, 2002.

Adam Kilgarriff and Gregory Grefenstette. Introduction to the special issue on the web as corpus. *Computational linguistics*, 29(3):333–347, 2003.

Oleksandr Kolomiyets and Marie-Francine Moens. A survey on question answering technology from an information retrieval perspective. *Information Sciences*, 181 (24):5412–5434, 2011.

G. Suresh Kumar and G. Zayaraz. Concept relation extraction using naã¯ve bayes classifier for ontology-based question answering systems. *Journal of King Saud University - Computer and Information Sciences*, 27(1):13 – 24, 2015. ISSN 1319-1578. doi: https://doi.org/10.1016/j.jksuci.2014.03.001. URL `http://www.sciencedirect.com/science/article/pii/S1319157814000020`.

Praveen Kumar, Shrikant Kashyap, Ankush Mittal, and Sumit Gupta. A query answering system for e-learning hindi documents. *South Asian Language Review*, 13(1&2):69–81, 2003.

Cody Kwok, Oren Etzioni, and Daniel S Weld. Scaling question answering to the web. *ACM Transactions on Information Systems (TOIS)*, 19(3):242–262, 2001.

Leah S Larkey, Margaret E Connell, and Nasreen Abduljaleel. Hindi clir in thirty days. *ACM Transactions on Asian Language Information Processing (TALIP)*, 2 (2):130–142, 2003.

Dominique Laurent, Patrick Séguéla, and Sophie Nègre. Qristal, système de questions-réponses. *TALN & RECITAL*, 1:53–62, 2005.

Wei Li and Andrew McCallum. Rapid development of hindi named entity recognition using conditional random fields and feature induction. *ACM Transactions on Asian Language Information Processing (TALIP)*, 2(3):290–294, 2003.

Xin Li and Dan Roth. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics, 2002a.

Xin Li and Dan Roth. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics, 2002b.

Xin Li and Dan Roth. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics, 2002c.

Xin Li and Dan Roth. Learning question classifiers: the role of semantic information. *Natural Language Engineering*, 12(03):229–249, 2006.

Ka Kan Lo and Wai Lam. Using semantic relations with world knowledge for question answering. In *TREC*, 2006.

Babak Loni. A survey of state-of-the-art methods on question classification. *Delft University of Technology, Tech. Rep*, 2011.

Babak Loni, Gijs Van Tulder, Pascal Wiggers, David MJ Tax, and Marco Loog. Question classification by weighted combination of lexical, syntactic and semantic features. In *International Conference on Text, Speech and Dialogue*, pages 243–250. Springer, 2011.

Vanessa Lopez, Victoria Uren, Marta Sabou, and Enrico Motta. Is question answering fit for the semantic web?: a survey. *Semantic Web*, 2(2):125–155, 2011.

Annie Louis and Ani Nenkova. A corpus of general and specific sentences from news. In *LREC*, pages 1818–1821, 2012.

Mark T Maybury. *New directions in question answering*. AAAI press, 2004.

David McDonald. Internal and external evidence in the identification and semantic categorization of proper names. *Acquisition of Lexical Knowledge from Text*, 1993.

Amit Mishra and Sanjay Kumar Jain. A survey on question answering systems with classification. *Journal of King Saud University-Computer and Information Sciences*, 28(3):345–361, 2016.

FA Mohammed, Khaled Nasser, and HM Harb. A knowledge based arabic question answering system (aqas). *ACM SIGART Bulletin*, 4(4):21–30, 1993.

Dan Moldovan and Mihai Surdeanu. On the role of information retrieval and information extraction in question answering systems. In *Information Extraction in the Web Era*, pages 129–147. Springer, 2002.

Dan Moldovan, Sanda Harabagiu, Marius Pasca, Rada Mihalcea, Roxana Girju, Richard Goodrum, and Vasile Rus. The structure and performance of an open-domain question answering system. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 563–570. Association for Computational Linguistics, 2000.

Dan Moldovan, Marius Paşca, Sanda Harabagiu, and Mihai Surdeanu. Performance issues and error analysis in an open-domain question answering system. *ACM Transactions on Information Systems (TOIS)*, 21(2):133–154, 2003.

Dan Moldovan, Christine Clark, Sanda Harabagiu, and Daniel Hodges. Cogex: A semantically and contextually enriched logic prover for question answering. *Journal of Applied Logic*, 5(1):49–69, 2007.

Aliod D. Moll. Qanswerfinder in trec 2003. In *TREC*, pages 392–398, 2003.

Diego Mollá and José Luis Vicedo. Question answering in restricted domains: An overview. *Computational Linguistics*, 33(1):41–61, 2007.

Diego Mollá, Menno van Zaanen, and Steve Cassidy. Named entity recognition in question answering of speech data. In *Proceedings of the Australasian Language Technology Workshop*, pages 57–65, 2007.

Christof Monz. *From document retrieval to question answering.* PhD thesis, Ph.D. thesis, University of Amsterdam, 2003.

Garima Nanda, Mohit Dua, and Krishma Singla. A hindi question answering system using machine learning approach. In *Computational Techniques in Information and Communication Technologies (ICCTICT), 2016 International Conference on*, pages 311–314. IEEE, 2016.

Elisa Noguera, Antonio Toral, Fernando Llopis, and Rafael Muńoz. Reducing question answering input data using named entity recognition. In *International Conference on Text, Speech and Dialogue*, pages 428–434. Springer, 2005.

Jong-Hoon Oh, Kentaro Torisawa, Chikara Hashimoto, Motoki Sano, Stijn De Saeger, and Kiyonori Ohtake. Why-question answering using intra-and inter-sentential causal relations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1733–1743, 2013.

Manuel Alberto Pérez-Coutiño, Manuel Montes-y Gómez, Aurelio López-López, and Luis Villaseñor Pineda. Experiments for tuning the values of lexical features in question answering for spanish. In *CLEF (Working Notes)*, 2005.

John Prager et al. Open-domain question–answering. *Foundations and Trends® in Information Retrieval*, 1(2):91–231, 2007.

Kumar P Praveen and Kiran V Ravi. Hybrid named entity recognition system for south and south east asian languages. In *Proceedings of the IJCNLP-08 Workshop on Named Entity Recognition for South and South East Asian Languages*, 2008.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, 2016.

Deepak Ravichandran and Eduard Hovy. Learning surface text patterns for a question answering system. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 41–47. Association for Computational Linguistics, 2002.

Santosh Kumar Ray, Amir Ahmad, and Khaled Shaalan. A review of the state of the art in hindi question answering systems. In *Intelligent Natural Language Processing: Trends and Applications*, pages 265–292. Springer, 2018.

Rami Reddy Nandi Reddy and Sivaji Bandyopadhyay. Dialogue based question answering system in telugu. In *Proceedings of the Workshop on Multilingual Question Answering*, pages 53–60. Association for Computational Linguistics, 2006.

Thomas Robb. Google as a corpus tool. *ETJ Journal*, 4(1):20–21, 2003.

Ian Roberts and Robert Gaizauskas. Evaluating passage retrieval approaches for question answering. In *European Conference on Information Retrieval*, pages 72–84. Springer, 2004.

Sandra Roger, Sergio Ferrández, Antonio Ferrández, Jesús Peral, Fernando Llopis, Antonia Aguilar, and David Tomás. Aliqan, spanish qa system at clef-2005. In *Workshop of the Cross-Language Evaluation Forum for European Languages*, pages 457–466. Springer, 2005.

Paolo Rosso, Yassine Benajiba, and Abdelouahid Lyhyaoui. Towards an arabic question answering system. In *Proc. 4th Conf. on Scientific Research Outlook & Technology Development in the Arab world, SROIV, Damascus, Syria*, pages 11–14, 2006.

Michael Rundell. The biggest corpus of all. *Humanising language teaching*, 2(3), 2000.

Sujan Kumar Saha, Sanjay Chatterji, Sandipan Dandapat, Sudeshna Sarkar, and Pabitra Mitra. A hybrid approach for named entity recognition in indian languages. In *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian languages*, pages 17–24, 2008.

Shriya Sahu, Nandkishor Vasnik, and Devshri Roy. Prashnottar: a hindi question answering system. *International Journal of Computer Science & Information Technology*, 4(2):149, 2012.

Tetsuya Sakai, Yoshimi Saito, Yumi Ichimura, Makoto Koyama, Tomoharu Kokubu, and Toshihiko Manabe. Askmi: A japanese question answering system based on semantic role analysis. In *Coupling approaches, coupling media and coupling languages for information retrieval*, pages 215–231. Le Centre de Hautes Etudes Internationales Dinformatique Documentaire, 2004.

Robert E Schapire. The strength of weak learnability. *Machine learning*, 5(2):197–227, 1990.

Nico Schlaefer. *A Semantic Approach to Question Answering*. AV Akademikerverlag, 2012.

Nico Schlaefer, Petra Gieselmann, Thomas Schaaf, and Alex Waibel. A pattern learning approach to question answering within the ephyra framework. In *International Conference on Text, Speech and Dialogue*, pages 687–694. Springer, 2006.

Satoshi Sekine and Ralph Grishman. Hindi-english cross-lingual question-answering system. *ACM Transactions on Asian Language Information Processing (TALIP)*, 2(3):181–192, 2003.

Khaled Shaalan. A survey of arabic named entity recognition and classification. *Computational Linguistics*, 40(2):469–510, 2014.

Padmaja Sharma, Utpal Sharma, and Jugal Kalita. Named entity recognition: A survey for the indian languages. *Parsing in Indian Languages*, page 35, 2011.

Cheng-Wei Shih, Min-Yuh Day, Tzong-Han Tsai, Tian-Jian Jiang, Chia-Wei Wu, Cheng-Lung Sung, Yu-Ren Chen, Shih-Hung Wu, and Wen-Lian Hsu. Asqa: Academia sinica question answering system for ntcir-5 clqa. In *NTCIR-5 Workshop, Tokyo, Japan*, pages 202–208, 2005.

Pushpraj Shukla, Amitabha Mukherjee, and Achla Raina. Towards a language independent encoding of documents. *NLUCS 2004*, page 116, 2004.

Joao Silva, Luísa Coheur, Ana Cristina Mendes, and Andreas Wichert. From symbolic to sub-symbolic information in question classification. *Artificial Intelligence Review*, 35(2):137–154, 2011.

John Sinclair et al. *Collins COBUILD English language dictionary*. Harper Collins Publishers,, 1987a.

John M Sinclair. *Looking up: An account of the COBUILD project in lexical computing and the development of the Collins COBUILD English language dictionary*. Collins Elt, 1987b.

Anil Kumar Singh. Named entity recognition for south and south east asian languages: Taking stock. In *IJCNLP*, pages 5–16, 2008a.

Anil Kumar Singh. Named entity recognition for south and south east asian languages: taking stock. In *Proceedings of the IJCNLP-08 Workshop on Named Entity Recognition for South and South East Asian Languages*, 2008b.

Tomek Strzalkowski and Sanda Harabagiu. *Advances in open domain question answering*, volume 32. Springer Science & Business Media, 2006.

Ang Sun, Minghu Jiang, Yifan He, Lin Chen, and Baozong Yuan. Chinese question answering based on syntax analysis and answer classification. *Acta Electronica Sinica*, 36(5):833–839, 2008.

Omar Trigui, Lamia Hadrich Belguith, and Paolo Rosso. Defarabicqa: Arabic definition question answering system. In *Workshop on Language Resources and Human Language Technologies for Semitic Languages, 7th LREC, Valletta, Malta*, pages 40–45, 2010.

Hiroshi Uchida and Meiying Zhu. The universal networking language beyond machine translation. In *International Symposium on Language in Cyberspace, Seoul*, pages 26–27, 2001.

Suzan Verberne, LWJ Boves, NHJ Oostdijk, and PAJM Coppen. Data for question answering: the case of why. 2006.

José L Vicedo, Ruben Izquierdo, Fernando Llopis, and Rafael Munoz. Question answering in spanish. In *Workshop of the Cross-Language Evaluation Forum for European Languages*, pages 541–548. Springer, 2003.

Ellen M Voorhees. Overview of the trec 2001 question answering track. *NIST special publication*, pages 42–51, 2002.

Ellen M Voorhees. Overview of trec 2003. In *Trec*, pages 1–13, 2003.

Ellen M Voorhees et al. The trec-8 question answering track report. In *TREC*, volume 99, pages 77–82, 1999.

Takahiro Wakao, Robert Gaizauskas, and Yorick Wilks. Evaluation of an algorithm for the recognition and classification of proper names. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pages 418–423. Association for Computational Linguistics, 1996.

David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

William A Woods. Progress in natural language understanding: an application to lunar geology. In *Proceedings of the June 4-8, 1973, national computer conference and exposition*, pages 441–450. ACM, 1973.

Hiroyasu Yamada, Taku Kudo, and Yuji Matsumoto. Japanese named entity extraction using support vector machine. *Transactions of Information Processing Society of Japan*, 43(1):44–53, 2002.

Zhang Yongkui, Zhao Zheqian, Bai Lijun, and Chen Xinqing. Internet-based chinese question-answering system. *Computer Engineering*, 15:34, 2003.

Zhou Yu, Jun Yu, Chenchao Xiang, Jianping Fan, and Dacheng Tao. Beyond bilinear: Generalized multimodal factorized high-order pooling for visual question answering. *IEEE Transactions on Neural Networks and Learning Systems*, (99): 1–13, 2018.

Dell Zhang and Wee Sun Lee. Question classification using support vector machines. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 26–32. ACM, 2003.

Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. Learning to count objects in natural images for visual question answering. *arXiv preprint arXiv:1802.05766*, 2018.

Zhiping Zheng. Answerbus question answering system. In *Proceedings of the second international conference on Human Language Technology Research*, pages 399–404. Morgan Kaufmann Publishers Inc., 2002a.

Zhiping Zheng. Answerbus question answering system. In *Proceedings of the second international conference on Human Language Technology Research*, pages 399–404. Morgan Kaufmann Publishers Inc., 2002b.