INTELLIGENT QUESTION ANSWERING ENGINE

Submitted by

S Ashish Bharadwaj	**12IT63**
Kalyanam Rajashree	**12IT28**
R Gokul Shanth	**12IT52**

Under the Guidance of

Prof. Ananthanarayana V. S.
Department of Information Technology,
NITK Surathkal, Mangalore

In partial fulfillment of the requirement for the award of the degree of

Bachelor of Technology

In

Information Technology

Department of Information Technology

National Institute of Technology Karnataka, Surathkal

2015-2016

Department of Information Technology
National Institute of Technology Karnataka Surathkal

<u>D E C L A R A T I O N</u>

We hereby declare that the Project Work Report entitled **Intelligent Question Answering Engine** which is being submitted to **The National Institute of Technology Karnataka, Surathkal** for the award of the degree of Bachelor of technology in **Information Technology**, is a bonafide report of the work carried out by us. The material content in this Project Work Report has not been submitted to any University or Institute for the award of any degree.

Name of the Student	Register No.	Signature with Date
Kalyanam Rajashree	12IT28	
R Gokul Shanth	12IT52	
S Ashish Bharadwaj	12IT63	

Place: NITK, Surathkal
Date:

Department of Information Technology
National Institute of Technology Karnataka Surathkal

C E R T I F I C A T E

This is to certify that the B.Tech. Project Work Report entitled **Intelligent Question Answering Engine** submitted by:

Kalyanam Rajashree	12IT28
R Gokul Shanth	12IT52
S Ashish Bharadwaj	12IT63

as the record of the work carried out by them, is accepted as the B.Tech. Project Work Report submission in the partial of the fulfillment of the requirements for the award of **Bachelor of Technology** in Information Technology.

Prof. Ananthanarayana V.S.
Project Guide, Dept of I.T,
NITK Surathkal, Mangalore

Prof. G. Ram Mohana Reddy
Chairman-DUGC, Dept of I.T,
NITK Surathkal, Mangalore

Abstract

Question Answering systems have been the topic of interest among researchers for a lot of years now. The field of question answering requires that the answer be precise, concise and retrieved quickly. Researchers have used various techniques to address these issues. In this project, we propose a hybrid approach to solve this problem combining the domains of information retrieval and natural language processing to answer factoid questions. We build a system which returns an answer sentence for any question posed in natural language. It is a double layered system with a document retrieval system working atop an answer sentence retrieval system. The answer sentence retrieval system works by predicting probability estimates for each question-sentence vector using a binary classifier. The system works on a pre-processed collection of documents stored on the local drive. By combining the best features from previous approaches and introducing a few of our own features, we are able to get better performance than the existing models. Additionally, we also build an intelligent prediction model which dynamically learns user patterns through multiple user interactions to predict what information the user might be looking for in the succeeding question. We also perform document clustering to reduce search space during document retrieval for a given query. The system's performance is measured using standard performance metrics, such as Mean Average Precision and Mean Reciprocal Rank, for information retrieval systems.

Keywords: *Question Answering, Information Retrieval, Natural Language Processing, Supervised Learning, Clustering*

Contents

1 Introduction **1**

2 Literature Survey **4**

2.1 Related Work 4

2.2 Outcome of Literature Survey 10

2.3 Problem Statement 11

2.4 Objectives 11

3 Methodology and Framework **13**

3.1 Question Processing and Classification 13

3.1.1 Building the Question Classifier 13

3.2 Document Processing 15

3.2.1 Tfidf Vectorization 15

3.2.2 Document Clustering 16

3.2.3 Topic Modeling and Topic Vectors 17

3.3 Document Retrieval 18

3.4 Answer Sentence Retrieval 19

3.4.1 Binary Classification 20

3.4.2 Features for Classifier 20

3.4.3 Performance Metrics 23

3.4.4 Answer Extraction Pipeline 23

3.5 Intelligent Answer Prediction 25

3.5.1 Weighted Edge Graph 25

3.5.2 Intelligent Answer Prediction Pipeline 25

4 Results and Analysis **28**
4.1 Dataset Description 28
4.2 Data Processing 29
4.3 Document Clustering 31
4.4 Intelligent Answer Prediction 31
4.5 User Interface 34
4.6 Question Classification 36
4.7 Document Retrieval 36
4.8 Answer Sentence Retrieval 36
4.9 End to End system 39

5 Conclusion and Future Work **42**

Bibliography **44**

Appendices **48**

A Turnitin Originality Report **49**

List of Figures

2.1.1 QA and Answer trigerring evaluation 6

3.3.1 Document Retrieval Pipeline 18

3.4.1 Logistic Regression Sigmoid Curve 21

3.4.2 Answer Sentence Retrieval Pipeline 24

3.5.1 Intelligent Answer Prediction Pipeline 26

4.1.1 Question Classes in WikiQA 29

4.1.2 WikiQA dataset statistics 29

4.3.1 Cluster Analysis 32

4.4.1 Weighted Edge Graph of ENTY 33

4.4.2 Weighted Edge Graph of ENTY at later state 33

4.4.3 Refresh Button in the UI 34

4.5.1 Screenshot of the User Interface 35

4.5.2 User Interface interacting with the Backend 35

4.6.1 Performance of the Question Classifier 36

4.7.1 Results for various queries 37

4.8.1 Graph of Performance of model on various question types 39

4.8.2 Performance for various Question types 39

4.9.1 Rank 1 answer with predictions under it 40

4.9.2 Rank 2 answer with predictions under it 40

4.9.3 Rank 3 answer with predictions under it 41

List of Tables

2.1 Comparison of Related Work 7

3.1 Question Class Table 14

4.1 Sample row from the Dataset 30
4.2 Feature Vector 30
4.3 Performance of proposed model compared to other models 38

Chapter 1

Introduction

Question Answering (QA) is a computer science discipline that deals with Natural Language Processing (NLP) and Information Retrieval (IR) in order to build systems capable of answering questions posed by the user in his natural language. With the amount of data available in the world growing at a massive scale and the speed of generation of this data also increasing with each day, it becomes crucial to device efficient QA systems. Such systems need to ensure that the user is provided with the right or most accurate answers in the least possible time. With the recent boom in Big Data and Distributed processing technologies the problem of QA has again taken the fore and new techniques to approach the problem are being devised. Over the past few years a lot of research has been going on in the field to improve and to better understand the natural language, grammar along with better ways to find answers which can be rephrased back to the natural language. Considering the huge amount of data that the user has to go through even after a successful search, it only makes sense to build systems that point the user to the exact location of the answer instead of a mere document suggestion. Recent research at Facebook, Google, IBM and such has proven that Question Answering is indeed still an active field calling out to active researchers to propose their ideas.

One of the most important QA types is answering factoid questions. Factual answer retrieval is supported by reasoning and evidence scoring methods. Fact based questions include questions such as, "When was the first cure for cancer made?", "Who discovered penicillin?", "Who is the lead actor in Adaptation?". The answers contain keywords aligning with who, what, where, when questions. Reasoning and list based questions are a much more complex question domain and require additional knowledge and faster processing to answer. An example of a list question would be ”How many rivers flow through Mysore?” from a document collection containing information about the geography of Karnataka. The system, in this case, would have to gather sentences relating to rivers and flowing through Mysore, then sum the numbers with the help of a list engine. Reasoning questions require the system to make deductions from the data that is present. Make inferences by establishing logic. This is a difficult type as building a reasoning engine requires a lot of processing power and answer retrieval time is high.

The QA task can be broadly classified into two, Closed Knowledge Base and Open Domain. In a Closed Domain system, questions can be asked and answers retrieved from a specific topic, for example, movies, arts, people, history,etc. Closed domain solutions are more robust as concrete grammar can be written which adheres to the logic of knowledge representation in that domain. Whereas in Open Domain Systems, techniques like computational linguistics, information retrieval from a set of relevant documents. These documents whether static or dynamic need to be pertained to a single specific domain or topic. Whereas, in an open domain setting, the documents can be from any domain and type.

There are various road blocks to be tackled when one aims to build a fully robust Question Answering system.

- Question Processing, Question classification and Context

  Numerous issues arise in question parsing. Same questions might have differing natural language representations. Classifying questions into various question classes is also difficult. Usually, questions are asked in context. It is important that the system is capable of analyzing the context before answer retrieval.

- Data Source

  A system can answer questions to which the facts are present in the knowledge base. The knowledge base has to be constructed for efficient retrieval.

- Real Time Answering, Answer extraction and ranking

  Answer generation has to be near real time due to the user’s immediate need for results. Also, optimal algorithms have to be implemented for answer retrieval which aid in quick and accurate results.

- Advanced and Interactive Reasoning

  Interactive systems need to be developed to better understand user questions. Interactive system architecture with effective reasoning schemes are a challenge to construct.

Progress has been made in terms of being able to answer questions based on facts, geospatial data, temporal and terminology. Active research is being carried out to this date to find better algorithms and improve search results and speed of retrieval.

Chapter 2

Literature Survey

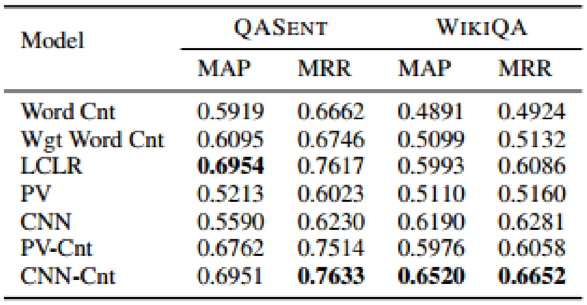
In this section, we will see some of the previous works in the field of Question and Answering(QA) and how researchers have tackled the problem and gather insights for us to arrive at our problem statement.

2.1 Related Work

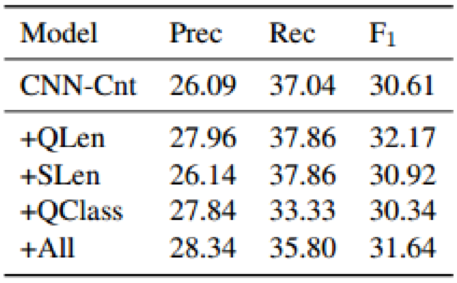
With technology pushing its way into everyday life, the need to develop intelligent QA bots which understand and respond in natural language that can assist humans is becoming very important. Question Answering is an age old problem of artificial intelligence. One of the earliest QA systems built was BASEBALL[20] in 1961 for baseball league based questions. Also, another popular work is that of the LUNAR system[21] developed in 1971, which enabled access to data about lunar rock and soil composition. Other early works in the field include SYNTHEX, LIFER,etc. The Text Retrieval Conference(TREC)[12] are a series of workshops aimed at progressing text retrieval techniques. It has a major QA track which led to big contributions in the field of QA research. Various algorithms were tested and their results tabulated. Although, the conferences led to a huge burst of research in the field, the techniques could not be effectively applied to real world text problems until recently.

Traditionally, there are three main approaches to solving this problem. These involve natural language processing, Information Retrieval and question template matching. NLP techniques involve deep parsing methods which involve detection of more relations between various components of a text, and not just the lexical role of words or phrases in a sentence. The approach involves making queries to structured semantic networks, extract information from the text, generate natural language responses. The process also involved conversion of text into a formal first order logic based triplet representation where it was ordered by subject-verb-object[13]. It involved constructing sophisticated semantic networks or knowledge graphs for relationship matching[23]. Information Retrieval proposes different methods for storage and access of information entities. IR systems are basically document retrieval systems which return related documents based on user queries. A more recent work in this field involves paragraph retrieval systems. The fundamental approach for QA Systems is to help achieve man-machine interaction and help retrieve specific answers to a question posed in natural language. Building a QA system requires information technology, artificial intelligence, knowledge, database management and cognitive science. Most of the QAS consist of three main modules, Question classification, Text or passage retrieval and answer processing. Answer processing is required so that answers are ranked so as to give the most relevant answer to the question rather than a vague one.[30][31][32]

WikiQA is a publicly available dataset that contains questions along with annotated sentences for open domain QA research. The dataset is constructed in a more natural process compared to other datasets such as the TREC. Yi yang et. al[1] compared systems on answer sentence selection and triggering using this dataset. Some of the systems used are Word count and weighted word count which use a non-stop word count and IDF respectively. It also includes two sentence semantic methods, Paragraph Vector and Convolution Neural Networks, Since MAP and MRR cannot be used as a measure for answer triggering, precision, recall and F1 are used at a question level. Only if a candidate answer crosses a certain threshold the answer is considered correct and selected. The MAP and MRR results can be seen on the table which compare two datasets(WikiQA and QASent). Similarly scores for answer triggering too can be seen below on the table.



Model	QASENT		WIKIQA	
	MAP	MRR	MAP	MRR
Word Cnt	0.5919	0.6662	0.4891	0.4924
Wgt Word Cnt	0.6095	0.6746	0.5099	0.5132
LCLR	**0.6954**	0.7617	0.5993	0.6086
PV	0.5213	0.6023	0.5110	0.5160
CNN	0.5590	0.6230	0.6190	0.6281
PV-Cnt	0.6762	0.7514	0.5976	0.6058
CNN-Cnt	0.6951	**0.7633**	**0.6520**	**0.6652**



Model	Prec	Rec	F_1
CNN-Cnt	26.09	37.04	30.61
+QLen	27.96	37.86	32.17
+SLen	26.14	37.86	30.92
+QClass	27.84	33.33	30.34
+All	28.34	35.80	31.64

Figure 2.1.1: QA and Answer trigerring evaluation

Wen-tau Yih et. al[31] focused on improving the performance using models of lexical semantic resources. Their MAP and MRR score showed a significant improvement of 8 to 10 points when compared to a baseline model. They formed pairs of words that were semantically related such as synonym/antonym, hypernymy/hyponymy. WordNet and thesauri form a great tool to cover synonyms, WordNet with a Is-A relations based in Probase is used to tackle Hypernymy and Hyponymy. Following the word-alignment paradigm, they find that the rich lexical semantic information improved the models consistently in the unstructured bag-of-words setting and also in the framework of learning latent structures.

Selecting answers from pre-retrieved sentences with the help of sequence tagging and Tree Edit Distance is what Xuchen Yao et. al[32] try to solve. Their model was free of manually created question and answer templates and yielded an F1 score of 63.3%. TED for sentence ranking saw a significant improvement is the answer selection by a 10% improvement in the F1 score in comparison to more standard features. This will improve and remove the necessity to make end to end implementations, this will help in improvement of QA systems by focusing more on other domains in the QA system

Table 2.1 has some of the pros and cons of related work done earlier.

Unlike previous tree-edit models which required a separate alignment-finding phase and resort to ad-hoc distance metrics, Mengqiu et. al[4] treated alignments as structured latent variables, and offers a principled framework for incorporating complex linguistic features. Their experiments on RTE and QA applications demonstrated that Tree-edit CRF models provide results competitive with previous syntax-based methods. Even with moderate improvements, their approach opened up a novel principal framework. It easily worked across different problem domains with minimum domain knowledge. But however,

Paper	Pros	Cons
Yih et. al	1. Using lexical Semantic resources 2. System is better regardless of algorithm 3. Outperforms syntactic analysis through dependency tree matching	1. Uncovered or inaccurate entity relations. 2. Lack of robust question analysis 3. The need of high-level semantic representation and inference
Mikolov et al.	1.Paragraph Vector learns fixed-length feature representations from variable-length pieces of texts 2. Overcome the weaknesses of bag-ofwords models 3. state-of-the-art results on several text classification and sentiment analysis tasks	1. Paragraph Vector can be expensive 2. Takes 30 minutes to compute the paragraph vectors test set, using a 16 core machine
Lei Yu et al.	1. Using Distributed representations to match questions with answers by considering their semantic encoding 2. Model easily applicable to a wide range of domains and languages 3. Model matches state of the art performance on the answer sentence selection task	1. Distributed representations are not very well equipped for dealing with cardinal numbers and proper nouns.
Callison-Burch et al.	1. Introduced novel features based on TED boosted F1 score by 10% compared with the use of more standard features 2. Modified design of the TED model is the state of the art in the task of ranking QA pairs	Without POS/NER/DEP features, F1 score drops 2. If CRF fails to find an answer, it is forced to tag an answer
Severyn et. al	1. Model is based on a convolutional neural network 2. A 3% absolute improvement in MAP and MRR2. 3. System requires no external parsers or resources.	1. Lower accuracy on long text pairs 2. strong overfitting on the training set will give poor generalization on the test data

Table 2.1: Comparison of Related Work

One of the biggest drawbacks was the lack of support for modeling phrasal re-ordering, which was a very common and important linguistic phenomena.

To efficiently extract sequences of edits, Michael at al.[5] employed a tree kernel as a heuristic in a greedy search routine. They described a logistic regression model that used 33 syntactic features of edit sequences to classify the sentence pairs. Experiments were conducted to evaluate tree edit models for three tasks: recognizing textual entailment, paraphrase identification, and an answer selection task for question answering. They offered an intuitive and effective method for modeling sentence pairs.

Heilman et al[6] builr on the idea that questions and their (correct) answers are related to each other via loose but predictable syntactic transformations. They proposed a probabilistic quasi-synchronous grammar, parameterized by mixtures of a robust non-lexical syntax/alignment model with a lexical-semantics-driven log-linear model. Their model significantly outperformed the two baseline algorithms even when they are given the benefit of WordNet on both development and test set, and on both MRR and MAP. Discriminative training and a relatively straightforward, barely engineered feature set were used in the implementation. Their scoring model was found to greatly outperform two state-of-the-art baselines on an answer selection task using the TREC dataset

When working with a free form factual question, constraints such as a semantic classification of the sought after answer and may even suggest using different strategies when looking for and verifying a candidate answer. Xin Li et. al[7] present the first work on a machine learning approach to question classification by developing a hierarchical classifier that classifies questions into fine-grained classes. It showed accurate results on a large collection of free-form questions used in TREC 10 and 11. Wen-tau Yih et. al[8] introduced a new vector space representation where antonyms lied on opposite sides of a sphere. This representation was derived with the aid of a thesaurus and latent semantic analysis (LSA). The key contribution of this work was to show how signs were assigned to the entries in the co-occurrence matrix on which LSA operated, so as to induce a subspace with the desired property. Improvement on the results were by 11 points absolute in F measure. Ming-Wei Chang et. al[9] proposed a general learning framework for a class of problems that required learning over latent intermediate representations. This paper developed a novel joint learning algorithm for both tasks, that uses the final prediction to guide the selection of the best intermediate representation. The algorithm was evaluated on three different NLP tasks transliteration, paraphrase identification and textual entailment and showed that their joint method significantly improves performance. Quoc Le et. al[10] propose Paragraph Vector, an unsupervised algorithm that learned fixed-length feature representations from variable-length pieces of texts, such as sentences, paragraphs, and documents. Their algorithm overcame the weakness bag-of-words models and also the results show that Paragraph Vectors outperformed them. With the help of this state of the art results were achieved on several text classification and sentient analysis tasks. Yu et. al[40] propose a novel approach to solving this task via means of distributed representations, and learn to match questions with answers by considering their semantic encoding. This contrasts prior work on this task, which typically relies on classifiers with large numbers of hand-crafted syntactic and semantic features and various external resources.

Analysis of the current challenge of knowledge management system is made by Hongming Zhu et al. [16] and then proposed graph based knowledge storage system which was multi-indexed to avoid the data duplication and optimized for parallel processing. Also, it explored the framework for the management, coding and analysis of large heterogeneous datasets.

This paper talks about QALC[24] and how it was used to improve gathering answers for questions from very selective sentences. The question required two kinds of answer, what to look for and where. The first part was solved by question analysis, where the expected answer was in the form of a named entity. The questions here too were made into triplets which helped in finding the location of the answers with the help of a list of associated templates.

Passage retrival was another important aspect in QAS, some researches like Hans Ulrich Christensen et al.[25] describe FuzzyPR, their passage retrieval system that applied fuzzy logic models in passage retrieval. Some of the main components that FuzzyPR included were automatic detection or term variants, proximity of question terms, fraction of question terms occurring in the passage. FuzzyPR achieved a higher coverage than Mean Reciprocal Rank. Ian Roberts et al.[26] focused on two new measures coverage and answer redundancy as they believed that it was more appropriate than the traditional recall and precision measures for overall QA performance. Xin Li et al.[27] proposed a passage-passage graph model as it can be used to improve the accuracy of relevance evaluation. Then the passage graph was made based on the similarities between the answers. This graph based passage ranking was used to re calculate the scores for the passages. This showed an improvement in the Mean Reciprocal Rank compared to other baseline methods.

David Ahn et al.[28] used Wikipedia as a source of answers for factoid questions and as an important model in detecting the material to be returned in response to questions. Their participation in the TREC 2004 QA track had convincing scores which led to further development of a more complex QA system with a credible source like Wikipedia.

Dynamic QA systems need automatic evaluation techniques, The NSIR[33] was one such system which used a set of metrics like First Hit Success, First Answer Reciprocal Word Rank, Total Reciprocal Rank and Precision. These metrics address the drawbacks of Mean Reciprocal Rank and are therefore more suitable for automatic evaluation of web based Question Answering Systems.

Rajendra Kumar et al.[34] propsed a new mechanism Tf-Idf based Apriori which clustered web documents. Each document in each cluster was ranked based on the Tf-idf and similarity factor based on the user query. This approach gave better results for higher minimum support with a good F-measure of 78%. This accuracy helped reduce the search space for a user query by getting the necessary documents in the beginning of every cluster.

The ever increasing RDF data was published as linked data, instinctive ways of accessing it becomes very important. Most Question answer systems translate questions into triplets of subject-verb-object which are paired or matched against the RDF data to get an answer. However, they believed that triplets do not represent an accurate representation on the semantic structure of the question in natural language. To overcome this, their approach included parsing the question to produce a SPARQL template that gave an insight into the internal structure of the question. This approach has helped them with many questions that could not be answered by other competing approaches.[35]

Big organizations such as IBM and Facebook with access to a lot of resources have the ability to make QAS on a large and complex scale. Watson is a Question Answering system developed by IBM specifically to take part in the show Jeopardy! Watson had access to around 200 million pages of data both in the form of structured and unstructured format along with the entire Wikipedia in text. It uses some techniques like natural language processing, information retrieval, knowledge representation, automated reasoning, and machine learning technologies to the field of open domain question answering.[37] The DeepQA[38] involves ranking all candidate answers according to their evidence scores and judging that each answer is correct. This is done using a machine learning framework that is phase-based, providing capabilities for manipulating the data and applying machine learning in successive applications. Scaling a QAS was done by facebook with the help of memory networks. They were able to answer simple questions, but given the large dataset this gives scope for scaling more complex reasoning and show that memory networks can be trained to achieve excellent results.[39]

2.2 Outcome of Literature Survey

Answer Sentence retrieval based Question Answering systems have been a field of interest for many years now. The field has gained popularity in the recent years due to the application of machine learning methods and the advent of faster processing systems. Answer sentence retrieval systems aim to return a ranked list of candidate answer sentences for a given question, from the document collection. Thus, there is a need to build a document retrieval system to reduce search space as the first level. Once the documents have been retrieved, the sentences can be extracted from the documents. Machine learning algorithms are used to get answer sentences from the retrieved documents. The methods thus used have explored a set of features pertaining to syntactic and semantic relationship between the question and the sentence. There is still a lot of scope for feature expansion. No earlier research has also worked in the direction of mapping the question domain to the answer domain.

Queries are natural language sentences describing the information that the user seeks. In information retrieval, a query need not uniquely be identified by a direct answer line in the vast text data. Instead, several documents may match the query, clearly with different degrees of relevancy. As the volume of data is increasing with each day, there is a need for faster processing architectures with efficient ranking methods. Hence, we propose our new architecture to try to solve this problem with more accurate answer retrieval. Our architecture also proposes an intelligence mechanism to the present domain using which, we can predict user's question types thereby speeding up information retrieval by early processing.

2.3 Problem Statement

The goal of the project is to build a robust and efficient Intelligent Question Answering Engine which is able to answer simple factoid questions by answer sentence extraction making use of a sentence retrieval system working under a document retrieval system.

2.4 Objectives

1. Question Processing and Classification

   The question input by the user has to be correctly broken down, unnecessary information removed, word tokenized and broken down into base forms. Additionally, the question needs to be classified using a pre-trained multi-class classifier into one of the six question types.

2. Document Processing

   The set of static documents have to be obtained from the data source. Then, similar text preprocessing steps as above have to be performed. The documents are then clustered based on word similarity so as to reduce document retrieval search space in the later steps.

3. Document Retrieval

   The pre-processed and clustered documents have tfidf vectors representing them. We vectorize the question based on the same vocabulary as the documents. We normalize these vectors to unit dimensions. Find cosine similarities and sort them to retrieve top relevant documents.

4. Answer Sentence Retrieval

   Once the question type is detected and relevant documents are retrieved, we need to rank the list of sentences that can possibly answer the question. We do this by training binary classifiers.

5. Interaction Management and Driver

   This is the main driver program controlling the entire architecture. It is responsible for the end to end performance of the system. It consists of a UI and a backend which work in coherence to record user interactions with the system on a weighted edge graph. The recorded user interactions are later used for answer prediction.

6. Intelligent Answer Prediction

   As user interactions with the system have been recorded, we build a model to predict the answer that the user might be looking for in the succeeding question. We perform a question type to answer domain mapping. We get to know the question type of the next question from the PFSA and then use this information to find the next answer sentence.

Chapter 3

Methodology and Framework

3.1 Question Processing and Classification

Question processing involves all the steps that are part of accepting a question from the user in natural language and processing it to produce machine understandable broken down pieces. The tasks involved in this step is some basic preprocessing like converting to lower case, removing punctuation, tokenizing the words. The words are further reduced to base form by stemming and lemmatization and removing stop words. Stop words are those words that occur frequently in the language. They need to be removed because they do not uniquely represent the sentence and are unimportant in our analysis.

We use these pre-built question classifier to classify the questions into the types Descriptive('DESC'), Entity('ENTY'), Numeric('NUM'), Location('LOC'), Human('HUM') and Abbreviation('ABBR'). We use the thus classified question type to extract further features.

3.1.1 Building the Question Classifier

Question Classification is an important task in the field of question answering. We need to know what type of a question the user has asked. Searching for an answer becomes easier when we know the type of question at hand. Questions of a certain type might have certain characteristics in the answers. Therefore, understanding question type will give cues to finding a better answer. We classify the questions into six question types - Descriptive('DESC'), Entity('ENTY'), Numeric('NUM'), Location('LOC'), Human('HUM') and Abbreviation('ABBR').

We build a logistic regression classifier for the same. We extract features from a given question which help us understand the question better and also allow the classifier to predict a class. As it is a multi-class classification problem with 6 classes, the classifier internally builds a 1 vs 5 classifier where it predicts the probabilities of a question being a given type or not. Then, it takes a maximum of these probabilities and assigns that class to the question.

The features that are extracted are both syntactic and semantic.

- **Syntactic**

  Syntactic features try to capture the grammar and structure of the language. The syntactic features which we use for this analysis are the words present in the sentence, the parts of speech tags present and chunks - non overlapping phrases in a sentence.

- **Semantic**

  Semantic features capture the meaning of the sentence. We extract the named entities present in the question and use this as a semantic feature to train the model. We also use the words present in these entities as features.

Using these features, we train a classifier using a labeled data set given by Li and Roth. The data set contains around 5500 labeled questions. The labels are the six question classes.

Table 3.1 contains question type and their examples.

Question Type	Question example
Descriptive(DESC)	What are liver enzymes?
Human(HUM)	Name the scar-faced bounty hunter of The Old West?
Numeric(NUM)	How many Jews were executed in concentration camps during WWII?
Abbreviation(ABBR)	What does YMCA stand for?
Entity(ENTY)	What is considered the costliest disaster the insurance industry has ever faced?
Location(LOC)	What sprawling U. S. state boasts the most airports?

Table 3.1: Question Class Table

3.2 Document Processing

Document processing involves preparing the documents for document retrieval. Firstly, the basic preprocessing steps have to be done. Stop words have to be removed. Sentences need to be converted to lower case, words tokenized, punctuation removed. Words are then lemmatized and stemmed to reduce them to the most base form. Then there are two kinds of processing that need to be done.

3.2.1 Tfidf Vectorization

Once the documents are pre-processed, tf-idf vectors of the documents need to be extracted. This is a vector representation of the bag of words present in the document. It is a bag of words approach and hence order of words is not taken into consideration. First, we go through all the documents in the collection and create a vocabulary by filtering out unnecessary words and numbers. This vocabulary serves as the dimensions for our tfidf vector. For every word in a document, tf represents the term frequency, which measures how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. Thus, the term frequency is often divided by the document length as a way of normalization. idf represents the inverse document frequency, which measures how important a term is. While computing TF, all terms are considered equally important. However it is known that certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance. Thus we need to weigh down the frequent terms while scale up the rare ones.

$$TF(t) = \frac{(Number\ of\ times\ term\ t\ appears\ in\ a\ document)}{(Total\ number\ of\ terms\ in\ the\ document)} \tag{3.1}$$

$$IDF(t) = log_e(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ with\ term\ t\ in\ it}) \tag{3.2}$$

tf-idf is then computed as the product of the two values for each word in the document. The tf-idf vectors are thuc computed for each document.

3.2.2 Document Clustering

Once we get the tf-idf vector representations of each of the documents, we cluster the documents into similar groups based on word similarity. Use KMeans algorithms to cluster the documents. k-means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. Given a set of observations (x1, x2, , xn), where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into k (n) sets S = S1, S2, , Sk so as to minimize the within-cluster sum of squares (WCSS) (sum of distance functions of each point in the cluster to the K center). In other words, its objective is to find:

$$argmin_s \sum_{i=1}^{k} \sum_{x \epsilon S_i}^{.} \|x - \mu_i\| \tag{3.3}$$

where u_i is the mean point of points in S_i

Measure effectiveness of clustering using internal clustering metrics. Since this is a purely unsupervised clustering scenario with no pre-decided labels for each document, it is only feasible to use internal metrics. We use silhouette score as the optimizing metric. We use silhouette score to find the optimal number of clusters. Assume the data have been clustered via any technique, such as k-means, into k clusters. For each datum i, let a(i) be the average dissimilarity of i with all other data within the same cluster. We can interpret a(i) as how well i is assigned to its cluster (the smaller the value, the better the assignment). We then define the average dissimilarity of point i to a cluster c as the average of the distance from i to points in c. Let b(i) be the lowest average dissimilarity of i to any other cluster, of which i is not a member. The cluster with this lowest average dissimilarity is said to be the "neighbouring cluster" of i because it is the next best fit cluster for point i. We now define a silhouette:

$$s(i) = \frac{b(i) - a(i)}{max\,\{a(i), b(i)\}} \tag{3.4}$$

We employ a hybrid approach to answer extraction. One approach uses a document indexed on our local storage for answer retrieval. The other method searches the web for related documents to find answer strings. The two methods work in parallel as represented in Fig 3.2. and results are later combined.

3.2.3 Topic Modeling and Topic Vectors

We introduce a new concept called average topic vector which will be explained in this section.

- **Topic Modeling**

  In natural language processing, Latent Dirichlet allocation (LDA) is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. For example, if observations are words collected into documents, it posits that each document is a mixture of a small number of topics and that each word's creation is attributable to one of the document's topics. LDA is an example of a topic model. For example, an LDA model might have topics that can be classified as CAT_related and DOG_related. A topic has probabilities of generating various words, such as milk, meow, and kitten, which can be classified and interpreted by the viewer as "CAT_related". Naturally, the word cat itself will have high probability given this topic. The DOG_related topic likewise has probabilities of generating each word: puppy, bark, and bone might have high probability. Words without special relevance, such as the (see function word), will have roughly even probability between classes (or can be placed into a separate category). A topic is not strongly defined, neither semantically nor epistemologically. It is identified on the basis of supervised labeling and (manual) pruning on the basis of their likelihood of co-occurrence. A lexical word may occur in several topics with a different probability, however, with a different typical set of neighboring words in each topic. Each document is assumed to be characterized by a particular set of topics. This is akin to the standard bag of words model assumption, and makes the individual words exchangeable.

- **Topic Vectors**

  We train LDA model on the list of sentences in the documents sentence wise. Then, we use the LDA model to predict the probability of a sentence belonging to a specific topic. Thus, if there are t topics. We get t probabilities. These t values are then normalized to unit vector dimension. This is the topic vector of a given sentence. This represents the topic distribution in the sentence.

- **Average Topic Vector**

From our training data set, we classify questions into the six question types. We separate out sentences corresponding to each of the question types. For each question type, we calculate the topic vectors for the sentences. Then, we separate the answer sentences from the non answer sentences. We then compute the average vectors of all the answer and non answer sentences. This vector upon normalization is called the average topic vector. The positive average vector of a certain question type represents a vector of the topic distributions in a sentence that correctly answers that question type. The negative average topic vector of a certain question type represents the vector of the topic distributions in a sentence that fails to answer a question of that question type correctly.

3.3 Document Retrieval

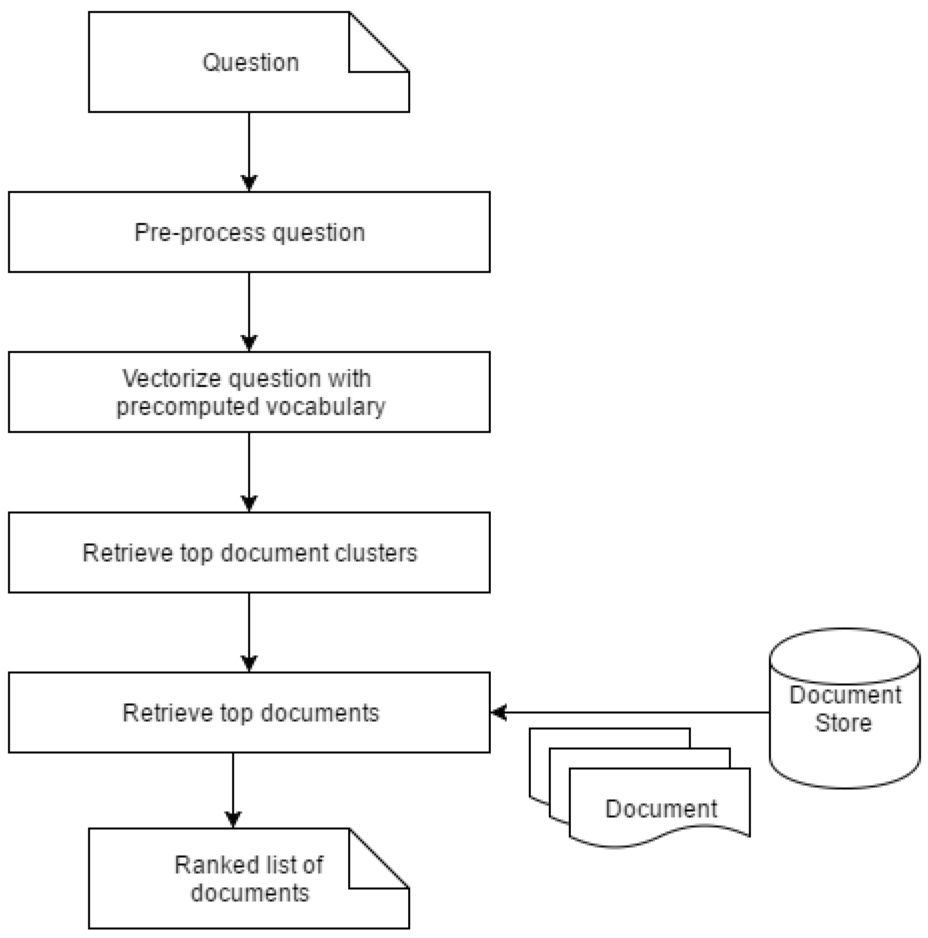


Figure 3.3.1: Document Retrieval Pipeline

Fig 3.3.1 details the steps involved in the document retrieval process

1. **Pre-process question**

   When a user enters a question, it has to be pre-processed as mentioned in the earlier section so that it is ready to be used for answer extraction.

2. **Vectorize the question**

   Use the vocabulary obtained from the document processing step to vectorize the question into a term frequency vector. The vector thus obtained can be further normalized to get a unit dimensional vector

3. **Retrieve top document clusters**

   Once the question vector is generated, we need to get the document clusters which are most relevant to the question. To do this, we get the cluster centers from the list of clusters. Cluster centers measure the tf-idf vector of the collection of documents in that cluster. Hence it collectively represents the bag of words of documents present in the cluster. We measure the cosine similarity of the cluster center vectors with the question vector. Cosine similarity is a measure of how close one vector is to the other. It takes the dot product of the two vectors divided by the magnitude of both of them. Once cosine similarity is found for all the cluster question pairs, we sort the clusters based on top scores. We then select the top n clusters and use it for the next step.

   $$CosineSimilarity([a,b,c],[x,y,z]) = \frac{ax+by+cz}{\sqrt{|a|^2+|b|^2+|c|^2} * \sqrt{|x|^2+|y|^2+|z|^2}} \tag{3.5}$$

4. **Retrieve top documents**

   Retrieve documents from the clusters returned in the earlier step. Get tfidf vectors of the documents thus returned. Similarly, measure cosine similarity and rank the documents. Retrieve top n documents and pass it to the answer sentence selection system.

3.4 Answer Sentence Retrieval

We use a carefully curated database of questions and answers to train models to answer questions. The dataset contains a question and a matching answer sentence amongst other such candidate answer sentences. We then extract features relating each question to a candidate answer sentence and mapping to its label. The dataset is divided into train, test and dev sets to experiment the building of the model. We use the Logistic Regression Classifier to build our model. Once the classifier model is built, it can successfully predict probability of label estimates for each of the question sentence pairs. We use this probability to rank the candidate answer sentences. Thus, a ranked list of answers is obtained. Similar to an Information Retrieval system, we use this ranked list of retrieved results, for each question, to measure the performance. As with information retrieval systems, we use popular metrics such as Mean Average Precision and Mean Reciprocal Rank to grade our system against other approaches.

3.4.1 Binary Classification

We use the Logistic Regression Classifier. The dependent variable is a categorical variable with binary output. We also require the system to generate an automated ranked list of sentences. Hence a logistic regression classifier which fits the training data onto a sigmoid function between 0 and 1 is the best. The sigmoid function values serve as probability estimates and can be used to further rank the retrieved sentences. The sigmoid function used in the logistic regression algorithm is shown below.

$$\sigma(t) = \frac{1}{1 + e^{-t}} \tag{3.6}$$

Where, t is the linear regression hypothesis function and the output of the above equation has a range of (0,1) by definition. It can also be seen clearly in Figure 3.4.1

3.4.2 Features for Classifier

From the question candidate sentence pairs, we extract a set of features after careful consideration of previous approaches and their performance measures.

- Unigram, Bigram and Trigram Count.

  A measure of the number of question unigrams, bigrams and trigrams present in the answer sentence. Before the token matching operation is performed, the question and sentence are both converted to lower case and stop words are removed. This feature serves as a measure of the degree of similarity between the question and the

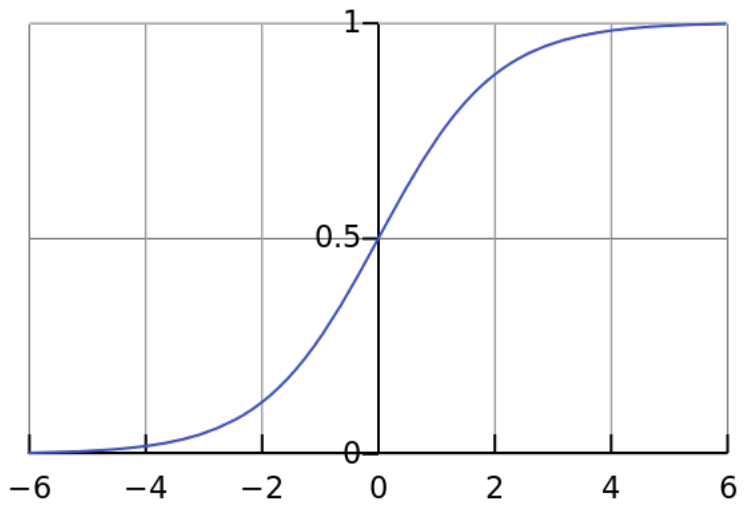


Figure 3.4.1: Logistic Regression Sigmoid Curve

sentence and allows for exploration of the importance of such a similarity in answer prediction.

- Weighted Word Count

  This feature is similar to the word count feature, except instead of adding a common sum of 1 every time a common word is found, we add the word's idf weight in the sentence set. This is a measure of how each word is uniquely important to the sentence at varying degrees. We use the IDF weighing scheme. IDF, or Inverse Document Frequency, is a measure of how uniquely a token is present in one document as opposed to other documents. It is given by the formula.

$$IDF = ln(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ containing\ the\ term\ t}) \tag{3.7}$$

- Lemma Matching

  We reduce each extracted unigram token, after stopword removal, to its base form by using lemmatization. This additional lemma matching step is done to capture the sentence matches that might have skipped the earlier steps.

- Wordnet Features - synonyms, antonyms

  Wordnet is an online database containing dictionary information. We use Wordnet to extract semantic features. We get synonyms and antonyms for all the words present in the question. Then the number of synonyms and antonyms present in the sentence is measure and a count is measured.

- Wordnet Features - hypernyms, hyponyms

  Hypernym is a super class of a given word. 'Vehicle' is hypernym of 'car'. Hyponym is a child class of the given word. These counts are extracted similar to the above step.

- Levenshtein Edit Distance

  In information theory and computer science, the Levenshtein distance is a string metric for measuring the difference between two sequences. Informally, the Levenshtein distance between two words is the minimum number of single-character edits (i.e. insertions, deletions or substitutions) required to change one word into the other. This feature is added to ensure a syntactic understanding of the question and the sentence. Mathematically, the Levenshtein distance between two strings a, b (of length $|a|$ and $|b|$ respectively) is given by $lev_{a,b}(|a|, |b|)$ where

$$lev_{a,b}(i,j) = \begin{cases} max(i,j) & if\ min(i,j) = 0, \\ min \begin{cases} lev_{a,b}(i-1,j)+1 \\ lev_{a,b}(i,j-1)+1 \\ lev_{a,b}(i-1,j-1)+1_{a_i \neq b_i} \end{cases} & otherwise \end{cases} \tag{3.8}$$

- Positive similarity and Negative Similarity
  We get the question type of the current question under consideration. We then retrieve the positive average vector and negative average vector for that question type. We also find the topic vector of the sentence with the pre-trained LDA model. We find the cosine similarity of the topic vector with the positive and negative average vectors. This is a measure of how close the sentence is to answer a question of the given type. Negative similarity is a measure of how far away the sentence is from answering a question of that question type. This is an important feature as it establishes a relationship between the question and sentence domain by linking the question type to the topics present in the answer.

3.4.3 Performance Metrics

1. Mean Reciprocal Rank

   Mean Reciprocal Rank is a measure of the effectiveness of an information retrieval system. The same measure can be used for a sentence retrieval based answer extraction system as both perform the same basic tasks. In each query, every sentence is assigned a rank based on the sorted probability estimates obtained after training the model. Mean Reciprocal Rank is the Average of the reciprocal of this rank over each query in the query set.

$$Mean\ Reciprocal\ Rank = \frac{1}{|Q|}\sum_{q=1}^{Q}\frac{1}{Rank\ of\ answer} \tag{3.9}$$

2. Mean Average Precision

   Like Mean Reciprocal Rank, Mean Average Precision is also a method used for measuring the performance of IR systems. But, it also serves as an excellent measure of evaluation QA systems. Average precision of a query is the average of precision measured at each of the true labels in the ranked list of sentences, i.e., it is the precision scores measured in a ranked list of documents at the positions of expected answers. Mean Average Precision is the mean of all these average precisions across all the queries.

$$Mean\ Average\ Precision = \frac{\sum_{q=1}^{Q} AvgPrecision(q)}{|Q|} \tag{3.10}$$

3.4.4 Answer Extraction Pipeline

The following steps are followed to get a ranked list of answers from a retrieved document set as shown in Fig 3.4.2

1. **Classify Question**

   Use the classifier built in the earlier step to classify the current question. The classification types are ENTY, DESC, NUM, HUM, LOC and ABBR which stand for entity, descriptive, numeric, human, location and abbreviation.

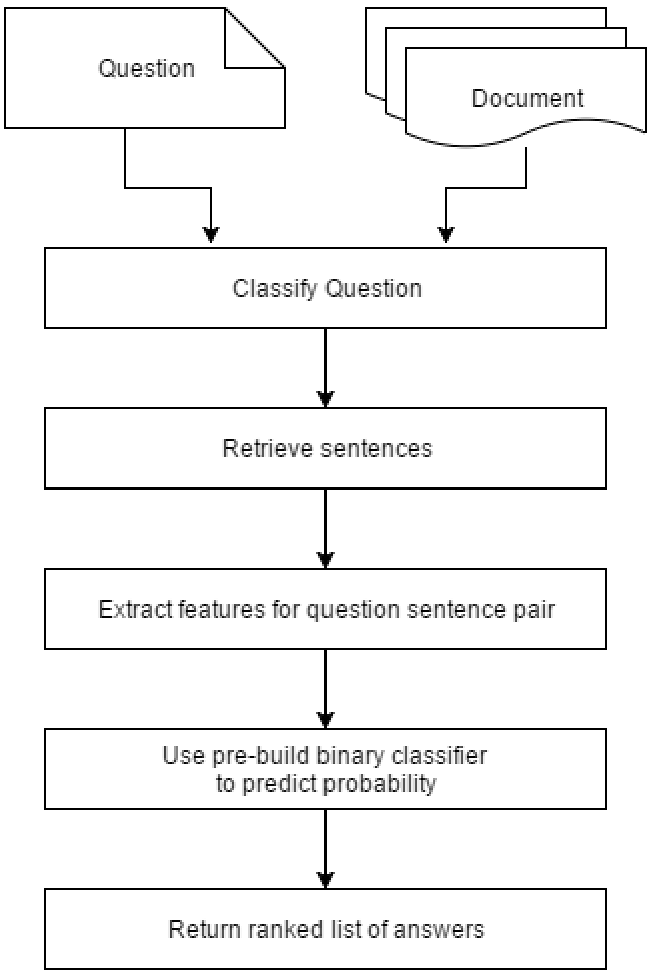

Question
Document
Classify Question
Retrieve sentences
Extract features for question sentence pair
Use pre-build binary classifier
to predict probability
Return ranked list of answers

Figure 3.4.2: Answer Sentence Retrieval Pipeline

2. **Retrieve Sentences**

   From the documents retrieved from the earlier step, extract the sentences and pre-process them. Create a dataset ready for further processing.

3. **Extract features for question-sentence pair**

   Extract the features mentioned in section 3.4.2. from the question and sentence pairs. This is a time consuming step as each row needs to processed independently. Thus, document retrieval step becomes essential.

4. **Use classifier to predict probability**

   Use the pre-built logistic regression classifier to predict the probability of each sentence being an answer to the given question. Since logistic regression is a probabilistic classifier it gives a probability estimate along with predicted class labels.

5. **Return ranked list of answers**

From the list of sentences that were considered from processing, return the top n sentences as answers for the question after sorting them based on prediction probabilities.

3.5 Intelligent Answer Prediction

We make use of user's previous interactions to build a profile for the user. Using this, we predict the probable next answer the user might be looking for.

3.5.1 Weighted Edge Graph

We build a weighted edge graph with the six question types as the nodes and the edges as transitions from one question to the next. An edge from a node A to node B contains the probability that the next question is of type B given that the current question is of type A. Thus, whenever a user enters a question, it is recorded in the database. The corresponding edge weights are updated. So, when a user asks a question, we know with a ranked list of probabilities the question type of the next question that the user will ask. This is a dynamic online learning approach as the graph gets updated with each user interaction with the system. For example, if the user has transitioned primarily to a DESC question after an ENTY type question, then the system tries to recommend the user with other DESC type questions in the document.

3.5.2 Intelligent Answer Prediction Pipeline

The following steps are to be taken to predict the next answer for a given user question. The Fig 3.5.1 describes the various steps.

1. **Get the next question type**
   Using the weighted edge graph, generate the next question type for a given question. The weighted edge graph gives a list of all the probabilities to all the six question types. Selecting the type with the maximum probability will give the next question type.

2. **Get the positive average topic vector**
   Once we get the question type of the next question, we then retrieve the positive

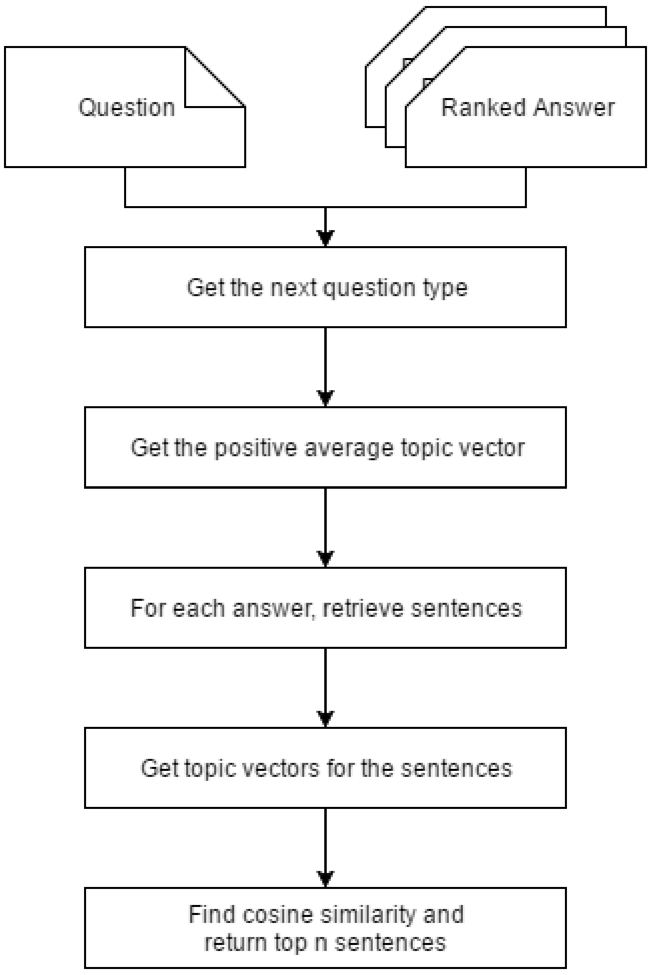

Question
Ranked Answer
Get the next question type
Get the positive average topic vector
For each answer, retrieve sentences
Get topic vectors for the sentences
Find cosine similarity and return top n sentences

Figure 3.5.1: Intelligent Answer Prediction Pipeline

average topic vector for that question type.

3. **For each answer, retrieve sentences**

   For each of the answers in the ranked list, retrieve the sentences from their corresponding documents. Thus if the system had returned n answers, we now have n documents for processing.

4. **Get topic vectors for sentences**

   For each of the above retrieved sentences, calculate topic vectors, which are probability estimates of the sentence belonging to any of the t topics in the LDA soft clustering.

5. **Find cosine similarity and return top n sentences**

   Find the cosine similarity between the topic vectors of the sentences and the positive average vector. Next, sort the sentences in order of decreasing cosine similarities and return the top n results.

Chapter 4

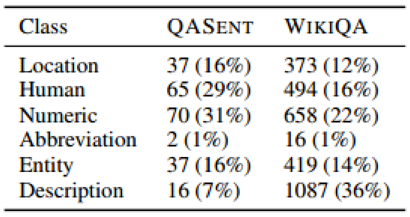
Results and Analysis

4.1 Dataset Description

For the answer sentence retrieval module, we use the WikiQA dataset. In order to reflect the true information need of common users, Bing query logs were used as the question source. The logs were taken during the period of May 1st, 2010 to July 31st, 2011 and queries were selected using simple heuristics, such as queries with a WH-word and queries ending with a question mark. Additionally, a few entity queries were also filtered out, such as the TV show "how i met your mother". In the end, approximately 2 percent of the queries were selected. To focus on factoid questions and to improve the question quality, only queries issued by at least 5 unique users were selected. a final lost of 3050 queries were sampled based on query frequencies.

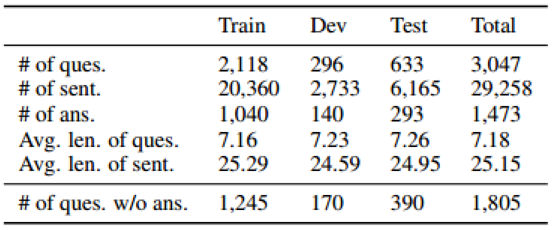
The WikiQA dataset is better than the QASent dataset in sheer size alone as it has more than ten times the number of questions. WikiQA dataset also contains a different composition of the different question types. It has a higher percentage of Description based question, which makes QA on the question set a more difficult task. The question classes can be seen in the Fig.4.1.1 . WikiQA data statistics showing the split of questions into train dev and test is shown in Fig.4.1.2

The data represented in one row of a dataset is shown in Table 4.1.



Class	QASent	WikiQA
Location	37 (16%)	373 (12%)
Human	65 (29%)	494 (16%)
Numeric	70 (31%)	658 (22%)
Abbreviation	2 (1%)	16 (1%)
Entity	37 (16%)	419 (14%)
Description	16 (7%)	1087 (36%)

Figure 4.1.1: Question Classes in WikiQA



	Train	Dev	Test	Total
# of ques.	2,118	296	633	3,047
# of sent.	20,360	2,733	6,165	29,258
# of ans.	1,040	140	293	1,473
Avg. len. of ques.	7.16	7.23	7.26	7.18
Avg. len. of sent.	25.29	24.59	24.95	25.15
# of ques. w/o ans.	1,245	170	390	1,805

Figure 4.1.2: WikiQA dataset statistics

4.2 Data Processing

The data is analyzed as a series of question and sentence pairs. The first step in the processing involves classifying the question at hand. The question is classified into one of the six question types. Then, the remaining features of word count, wordnet features, edit distance and positive and negative similarity are obtained. The features thus obtained are used for training the logistic regression classifier using the answer labels present in the train set. Once the models have been built on the train set, we test the model performance on test and dev sets using MAP and MRR scores. We also perform question type wise performance analysis to find out how the model performs with each question type. The results are tabulated and visualized.

The feature vector of one row after all the processing can be seen in the Table 4.2.

QuestionID	Q1
Question	how are glacier caves formed?
DocumentID	D1
DocumentTitle	Glacier cave
SentenceID	D1-0
Sentence	A glacier cave is a cave formed within the ice of a glacier.
Label	1
Type	train

Table 4.1: Sample row from the Dataset

Unigram	2
Bigram	0
Trigram	0
Lemma Count	3
W-idf	1.832581464
Synonym	0
Antonym	0
Hypernym	0
Hyponym	0
Levenshtein distance	4
Positive Similarity	0.479220407
Negative Similarity	0.395776447
Question Type	DESC
Question	how are glacier caves formed
Sentence	a glacier cave is a cave formed within the ice of a glacier

Table 4.2: Feature Vector

4.3 Document Clustering

We get the tf-idf vectors for the contents of the 3000 documents in the document store. Once we have vectors for each of the documents, we also get the vocabulary that define the dimensions of the vector. Now we cluster the documents using K-Means clustering. One drawback of KMeans clustering is that the user has to specify the number of clusters K. In our case, since there are 3000 documents, any number of clusters which would reduce the average number of documents per cluster to around 200 is good. So, we try to fit the K Means model with different values for K ranging between 5 and 20. We measure the silhouette score for each of these clustering methods. Silhouette score gives an estimate of the effectiveness of the clustering algorithm. Once we get the silhouette scores of all of the models with varying values of K, we plot them to get the best silhouette score and thereby select the best model. In our analysis, as can been seen in the Fig 4.3.1. below, the best KMeans model was for K = 15 and thus we have chosen the number of clusters to be 15 in our study.

4.4 Intelligent Answer Prediction

We predict the question type of the user's next question using user's history of question logs. This is a dynamically learning model. We have a weighted edge graph where each node is a question type. When a user asks a question, the question type of that question is found and recorded in the system. When the user asks another question, then the question type of that question is found and the edge connecting the first type to the second type is incremented by 1.

Fig 4.4.1 represents the weighted edge graph of the ENTY question type. The nodes are the question type while the edges signy the trasition from one question type to the other. Inititally in the figure, the numbers mean that the user has trasitioned that many times in the direction of the edge. For example, The user has transitioned 7 times from ENTY to DESC, 5 times from ENTY to ENTY, 4 rimes from ENTY to NUM and so on. Suppose a user interacts with the system asking it ENTY questions 2 times and DESC questions 3 times after asking an ENTY question, then the updated weighted edge graph at a later time would look like Fig 4.4.2. where the edge between ENTY to DESC which was previously 7 is now 10 and similarly the one between ENTY to ENTY which was 5

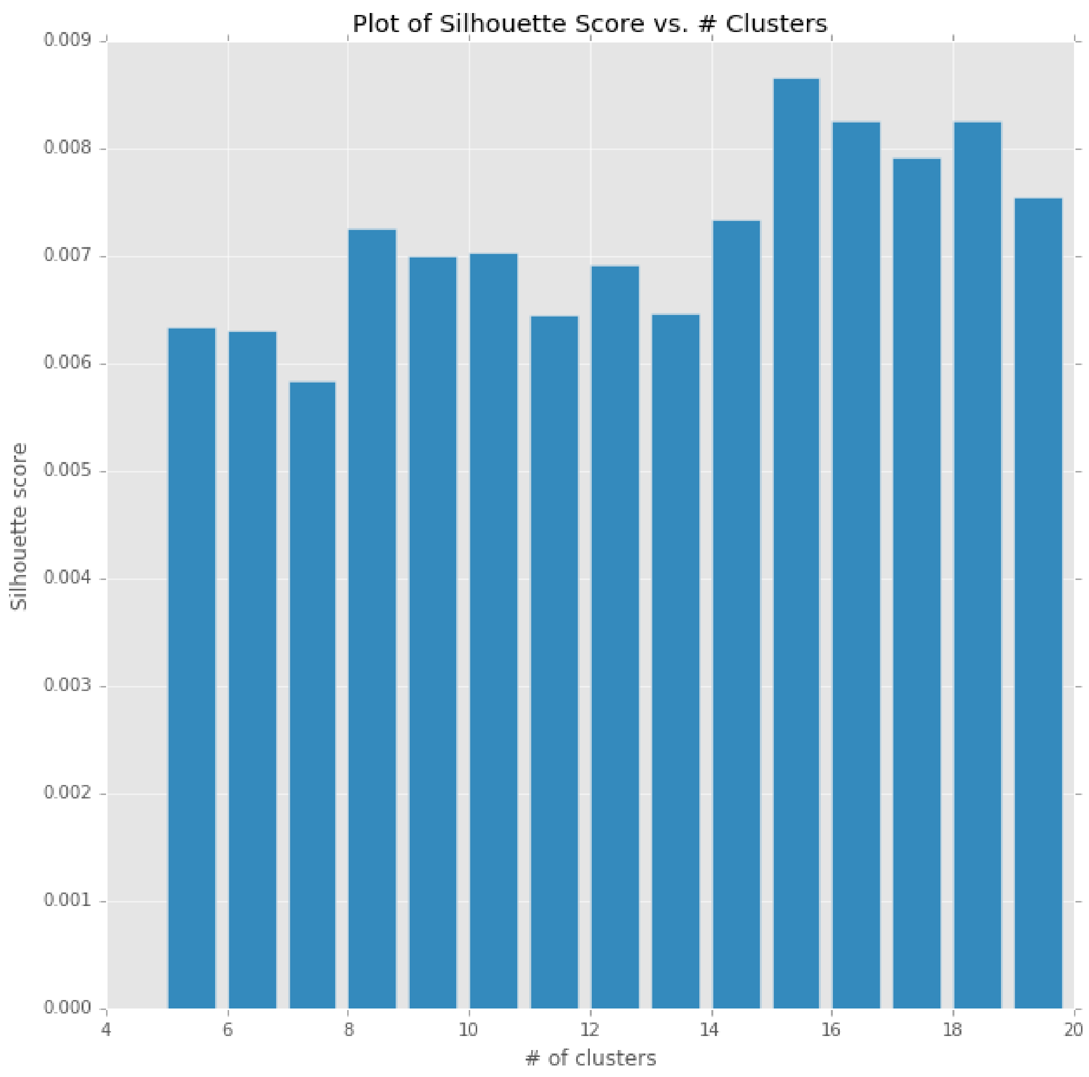


Figure 4.3.1: Cluster Analysis

is now 7.

Now, in order to predict the user's next question type, we go to the node of the user's current question's question type. Get the edge weights of all the edges to different question types. Then find the maximum of the values. This is the edge that has the highest probability of happening. It means that the next question that is asked by the user will have a transition over this edge.

Additionally, we have a REFRESH functionality built into our system. As can be seen in the Fig 4.4.3. Whenever a user wants to start a new series of questions, he can hit the refresh button. The next question he enters will be considered a new question and no update will be made in the weighted edge graph. This feature allows the user to effectively transition from one kind of search to another without hindering the weighted

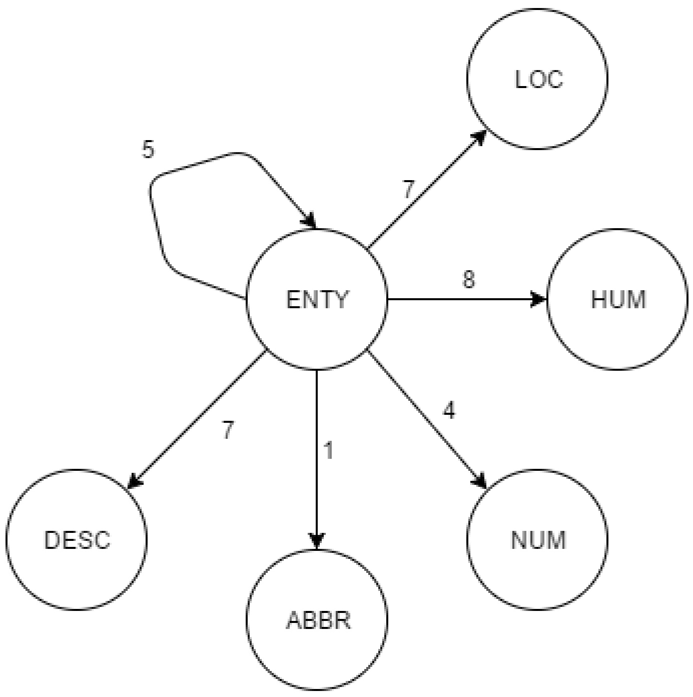


Figure 4.4.1: Weighted Edge Graph of ENTY

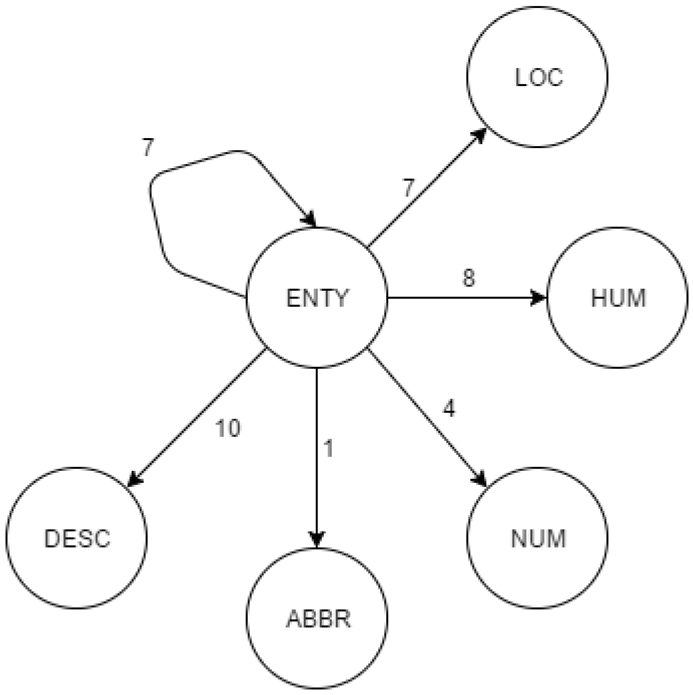


Figure 4.4.2: Weighted Edge Graph of ENTY at later state

edge graph mechanism.

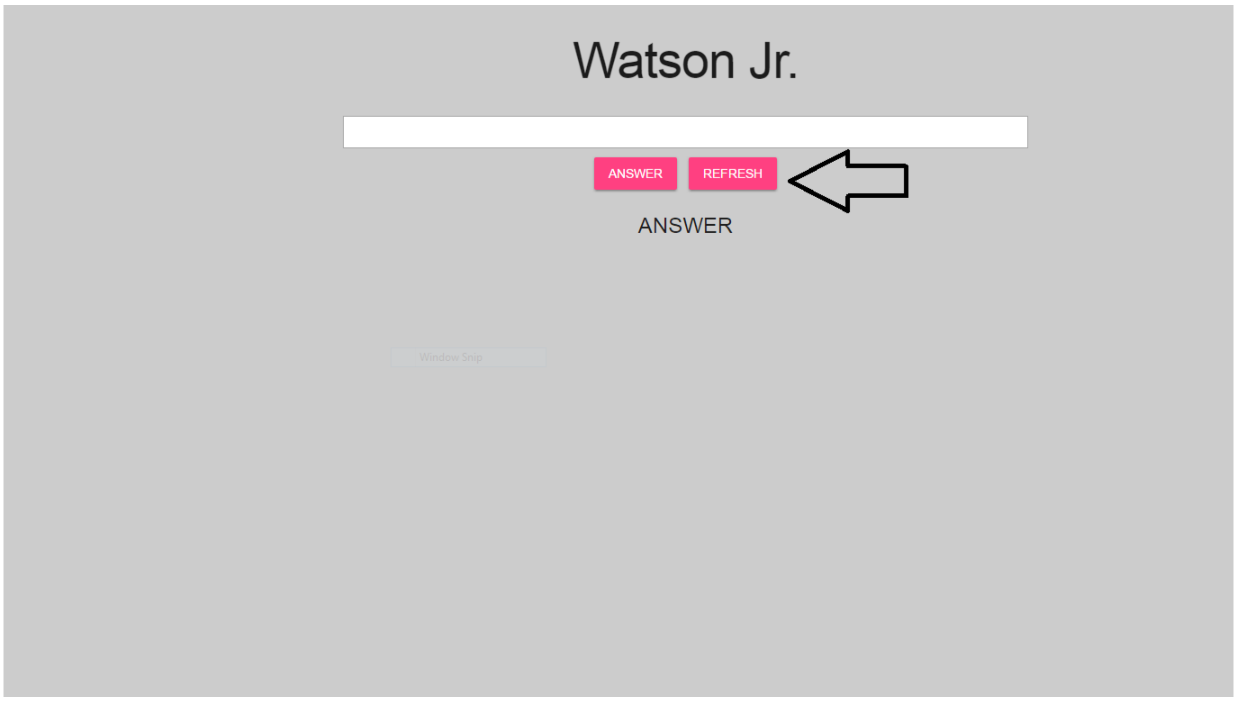
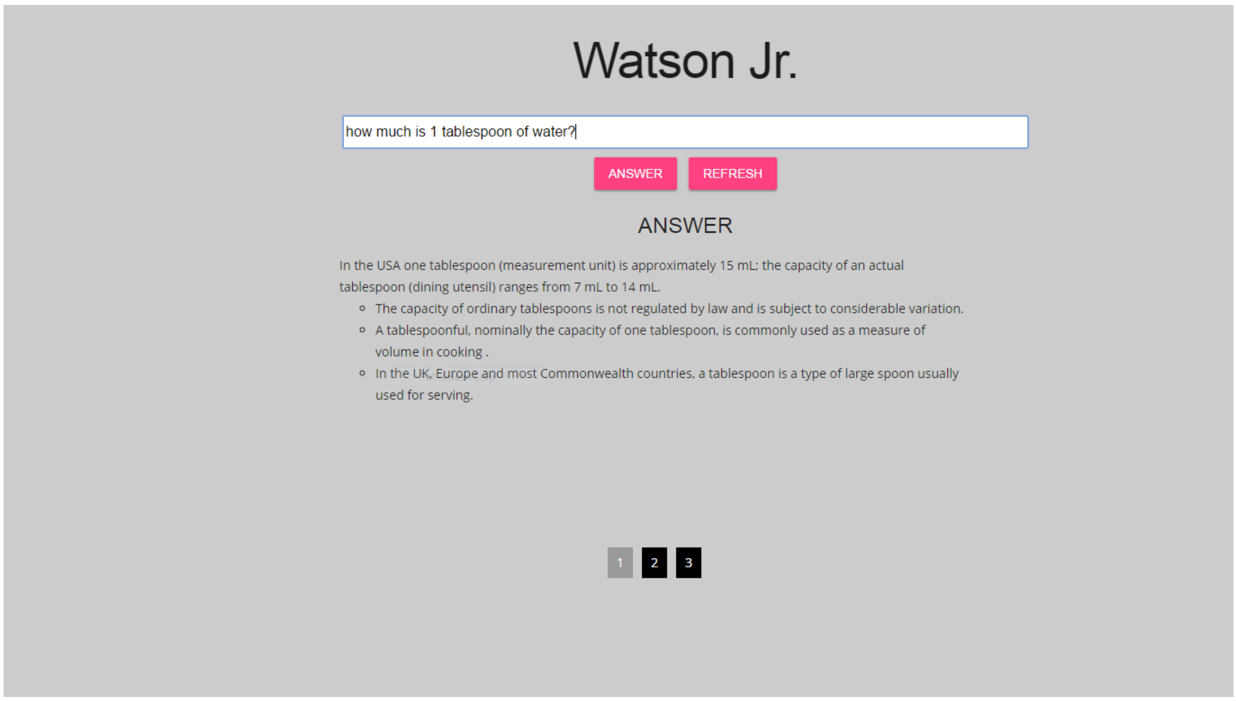


Figure 4.4.3: Refresh Button in the UI

4.5 User Interface

The user interface is built using python, HTML, CSS, Javascript and Jquery. The UI consists of a search box for the user to enter his question. The UI offers two buttons - ANSWER and REFRESH. Answer makes a HTTP request to the backend and fetches the answer. The backend consists of a python flask containing the code responsible for fetching the answer to a given question. The entire pipeline described in the earlier sections is performed. A ranked list of answers is returned. For each answer, there are 3 prediction answers. The UI is designed in such a way that there is a slider interface for displaying the answers. The prediction answers are displayed under each answer as an unordered list. Fig 4.5.1 shows the User Interface for some user query. Fig 4.5.2 shows the terminal while the flask program is running in the background. One can see the GET requests being made to the flask code from the UI. The top documents retrieved are also seen in the image.

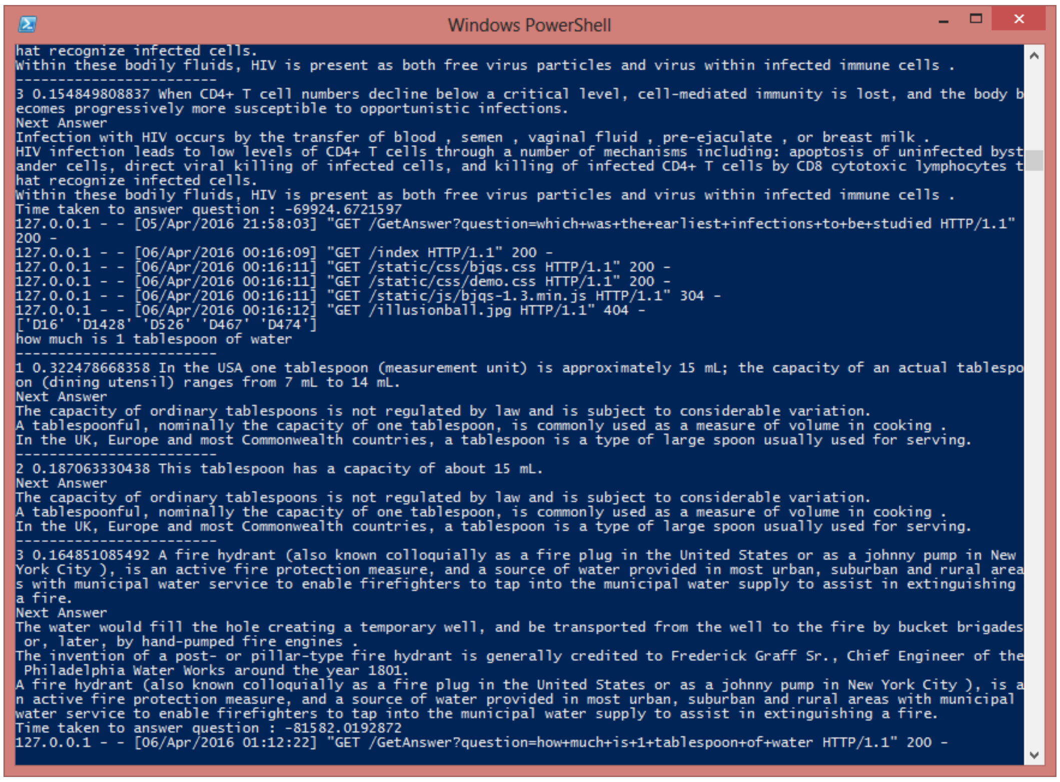


ANSWER

In the USA one tablespoon (measurement unit) is approximately 15 mL; the capacity of an actual tablespoon (dining utensil) ranges from 7 mL to 14 mL.

- The capacity of ordinary tablespoons is not regulated by law and is subject to considerable variation.
- A tablespoonful, nominally the capacity of one tablespoon, is commonly used as a measure of volume in cooking .
- In the UK, Europe and most Commonwealth countries, a tablespoon is a type of large spoon usually used for serving.

Figure 4.5.1: Screenshot of the User Interface

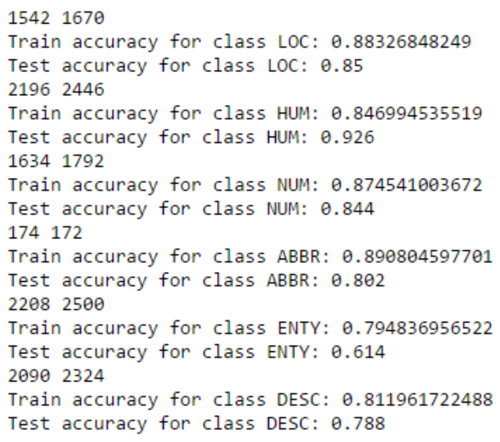


```
hat recognize infected cells.
Within these bodily fluids, HIV is present as both free virus particles and virus within infected immune cells .
------------------------
3 0.154849808837 When CD4+ T cell numbers decline below a critical level, cell-mediated immunity is lost, and the body b
ecomes progressively more susceptible to opportunistic infections.
Next Answer
Infection with HIV occurs by the transfer of blood , semen , vaginal fluid , pre-ejaculate , or breast milk .
HIV infection leads to low levels of CD4+ T cells through a number of mechanisms including: apoptosis of uninfected byst
ander cells, direct viral killing of infected cells, and killing of infected CD4+ T cells by CD8 cytotoxic lymphocytes t
hat recognize infected cells.
Within these bodily fluids, HIV is present as both free virus particles and virus within infected immune cells .
Time taken to answer question : -69924.6721597
127.0.0.1 - - [05/Apr/2016 21:58:03] "GET /GetAnswer?question=which+was+the+earliest+infections+to+be+studied HTTP/1.1"
200 -
127.0.0.1 - - [06/Apr/2016 00:16:09] "GET /index HTTP/1.1" 200 -
127.0.0.1 - - [06/Apr/2016 00:16:11] "GET /static/css/bjqs.css HTTP/1.1" 200 -
127.0.0.1 - - [06/Apr/2016 00:16:11] "GET /static/css/demo.css HTTP/1.1" 200 -
127.0.0.1 - - [06/Apr/2016 00:16:11] "GET /static/js/bjqs-1.3.min.js HTTP/1.1" 304 -
127.0.0.1 - - [06/Apr/2016 00:16:12] "GET /illusionball.jpg HTTP/1.1" 404 -
['D16' 'D1428' 'D526' 'D467' 'D474']
how much is 1 tablespoon of water
------------------------
1 0.322478668358 In the USA one tablespoon (measurement unit) is approximately 15 mL; the capacity of an actual tablespo
on (dining utensil) ranges from 7 mL to 14 mL.
Next Answer
The capacity of ordinary tablespoons is not regulated by law and is subject to considerable variation.
A tablespoonful, nominally the capacity of one tablespoon, is commonly used as a measure of volume in cooking .
In the UK, Europe and most Commonwealth countries, a tablespoon is a type of large spoon usually used for serving.
------------------------
2 0.187063330438 This tablespoon has a capacity of about 15 mL.
Next Answer
The capacity of ordinary tablespoons is not regulated by law and is subject to considerable variation.
A tablespoonful, nominally the capacity of one tablespoon, is commonly used as a measure of volume in cooking .
In the UK, Europe and most Commonwealth countries, a tablespoon is a type of large spoon usually used for serving.
------------------------
3 0.164851085492 A fire hydrant (also known colloquially as a fire plug in the United States or as a johnny pump in New
York City ), is an active fire protection measure, and a source of water provided in most urban, suburban and rural area
s with municipal water service to enable firefighters to tap into the municipal water supply to assist in extinguishing
a fire.
Next Answer
The water would fill the hole creating a temporary well, and be transported from the well to the fire by bucket brigades
 or, later, by hand-pumped fire engines .
The invention of a post- or pillar-type fire hydrant is generally credited to Frederick Graff Sr., Chief Engineer of the
 Philadelphia Water Works around the year 1801.
A fire hydrant (also known colloquially as a fire plug in the United States or as a johnny pump in New York City ), is a
n active fire protection measure, and a source of water provided in most urban, suburban and rural areas with municipal
water service to enable firefighters to tap into the municipal water supply to assist in extinguishing a fire.
Time taken to answer question : -81582.0192872
127.0.0.1 - - [06/Apr/2016 01:12:22] "GET /GetAnswer?question=how+much+is+1+tablespoon+of+water HTTP/1.1" 200 -
```

Figure 4.5.2: User Interface interacting with the Backend

4.6 Question Classification

We build a multi(6) class question classifier on a dataset containing 5500 labeled questions. The Fig 4.6.1. details the percentage accuracy obtained with each of the question classes. As it can be seen from the image, the accuracy is quite high and thus can be used a classifier for classifying questions of our model.



```
1542 1670
Train accuracy for class LOC: 0.88326848249
Test accuracy for class LOC: 0.85
2196 2446
Train accuracy for class HUM: 0.846994535519
Test accuracy for class HUM: 0.926
1634 1792
Train accuracy for class NUM: 0.874541003672
Test accuracy for class NUM: 0.844
174 172
Train accuracy for class ABBR: 0.890804597701
Test accuracy for class ABBR: 0.802
2208 2500
Train accuracy for class ENTY: 0.794836956522
Test accuracy for class ENTY: 0.614
2090 2324
Train accuracy for class DESC: 0.811961722488
Test accuracy for class DESC: 0.788
```

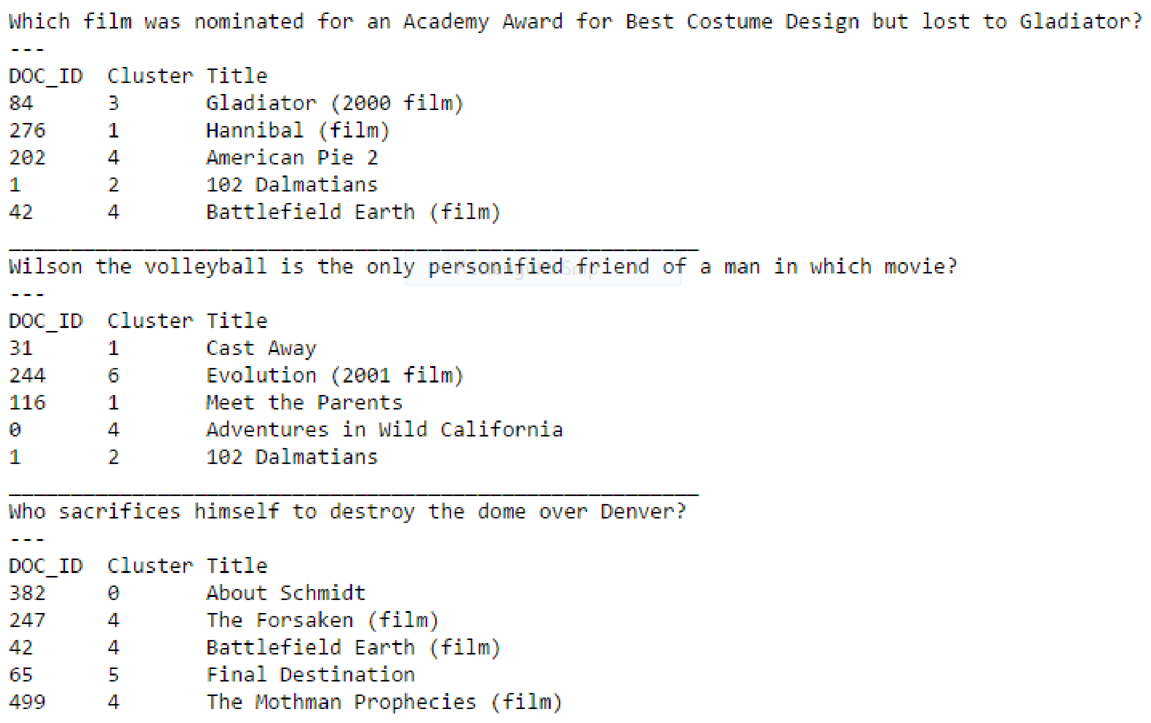
Figure 4.6.1: Performance of the Question Classifier

4.7 Document Retrieval

Fig 4.7.1 shows the results returned for various queries. It contains the top five matching documents for each of the three queries and also the clusters to which the documents belong to. It can be seen that the answer is present in 4, 1 and 3 ranks respectively. Thus, the document retrieval is efficient as the correct document needed to answer the query is within top 5 documents. Hence we use only the top 5 documents for our sentence retrieval system.

4.8 Answer Sentence Retrieval

Once models were built on the mentioned features, we tested it on dev and test sets and measured the scores of Mean Average Precision and Mean Reciprocal Rank. We obtained MAP and MRR scores of 0.690 and 0.678 respectively. As can be seen from the Table 4.3 , our model clearly has a better performance than any other model until now when tested on this dataset. This can be accounted for the fact that our model has a wider spectrum of



```
Which film was nominated for an Academy Award for Best Costume Design but lost to Gladiator?
---
DOC_ID  Cluster Title
84      3       Gladiator (2000 film)
276     1       Hannibal (film)
202     4       American Pie 2
1       2       102 Dalmatians
42      4       Battlefield Earth (film)
________________________________________________________
Wilson the volleyball is the only personified friend of a man in which movie?
---
DOC_ID  Cluster Title
31      1       Cast Away
244     6       Evolution (2001 film)
116     1       Meet the Parents
0       4       Adventures in Wild California
1       2       102 Dalmatians
________________________________________________________
Who sacrifices himself to destroy the dome over Denver?
---
DOC_ID  Cluster Title
382     0       About Schmidt
247     4       The Forsaken (film)
42      4       Battlefield Earth (film)
65      5       Final Destination
499     4       The Mothman Prophecies (film)
```

Figure 4.7.1: Results for various queries

features that it works on. Earlier systems have either focused entirely on either syntactic or semantic features without giving any regard to striking an effective balance between the two. The LCLR system uses parse tree edit distances between question and sentence to compute models to predict the answer sentence. In our method, we make use of a similar edit distance method but additionally, we also use other semantic features also. The semantic features such as lemma count, wordnet synonyms, antonyms, hypernyms and hyponyms help add a sense of vocabulary understanding to the system and help better its performance. We also have word counts of unigram, bigram and trigrams in the question sentence pair. This ensures we get basic string matching features too. Thus with the combination of syntactic, semantic and string matching features we were able to get an accuracy equal to that of the current models. We then introduced two new features - positive similarity and negative similarity. By building a topic model on the sentences data, we are able to generate a vector representation of the distribution of various topics in the sentence. We establish a link between the question type and the topic distributions. Basically, this is a measure of how well a sentence can answer a question of that question type. Thus by including this feature, we were able to find the internal characteristics of a sentence that answers a question of a certain question type.

Model	**WikiQA**	
	MAP	**MRR**
Word Cnt	0.4891	0.4924
Wgt Word Cnt	0.5099	0.5132
LCLR	0.5993	0.6086
PV	0.5110	0.5160
CNN	0.6190	0.6281
PV-Cnt	0.5976	0.6058
CNN-Cnt	0.6520	0.6652
Proposed Method(Dev)	**0.6810**	**0.6858**
Proposed Method(Test)	**0.6789**	**0.6900**

Table 4.3: Performance of proposed model compared to other models

The table 4.8.1 and figure 4.8.2 show the performance of our model with respect to various question types. The six bars show the MAP and MRR scores for the dev, test and train datasets respectively. There are six types of questions, namely, ENTY, DESC, LOC, HUM, NUM and ABBR. As ABBR type questions were absent from the dataset, we have removed them from the plot. The graph contains the plot for the six scores mentioned above for each of the five question types. We can see that our model performs extremely well for LOC questions with MAP scores of 0.76. The model also performs really well for ENTY based questions with an MAP score of 0.73. The model performs as good as the best models in the world. For DESC questions the model underperforms. This can be accounted for the fact that descriptive questions have very little information associated with them and have a wide range of answer sentence characteristics which is difficult to mine. Researchers all over the world have fallen short of successfully being able to answer DESC questions. Our model only aims at answering simple factoid questions and those question types, it answers very effectively.

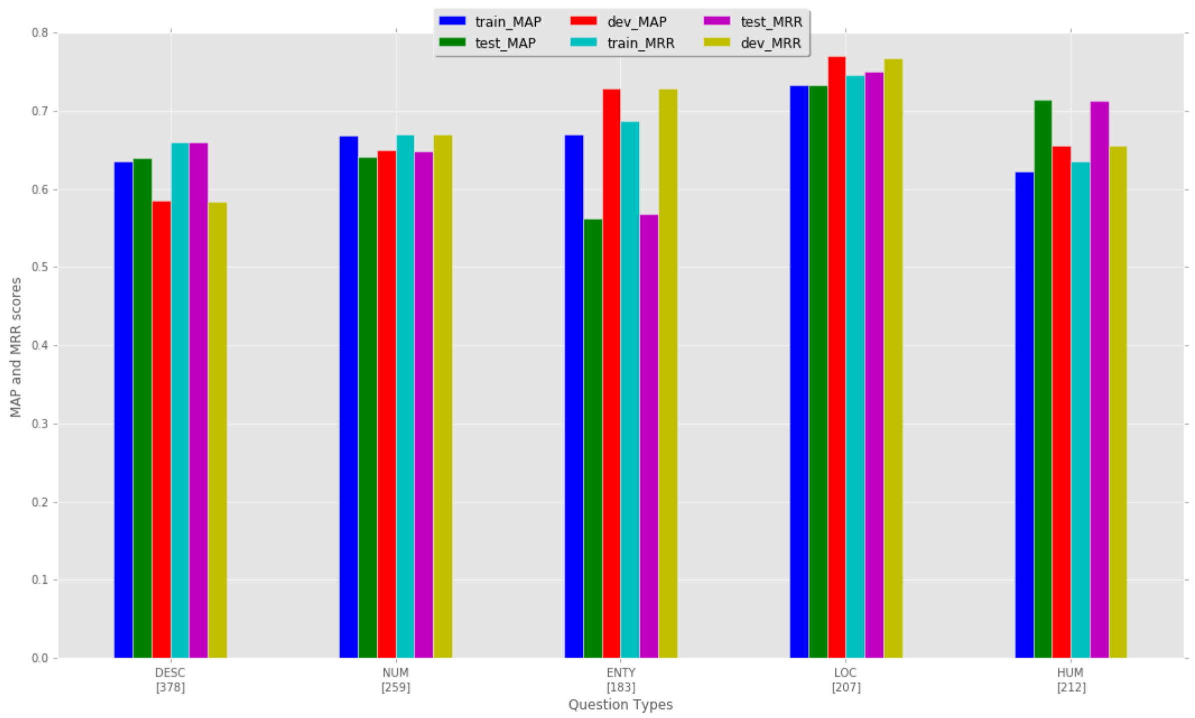
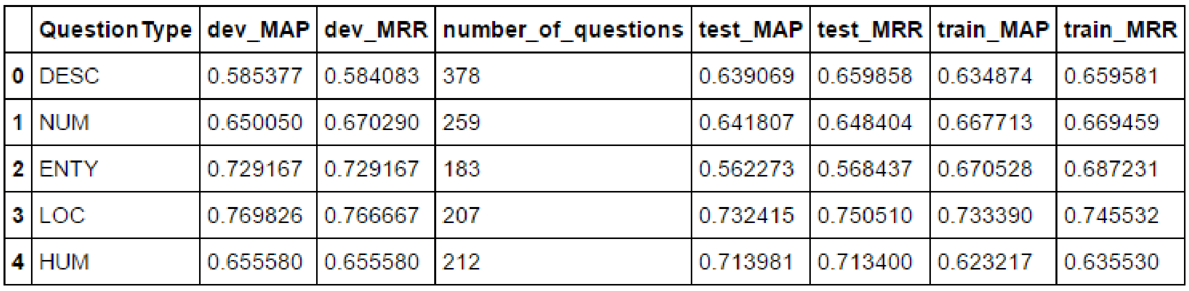


Figure 4.8.1: Graph of Performance of model on various question types

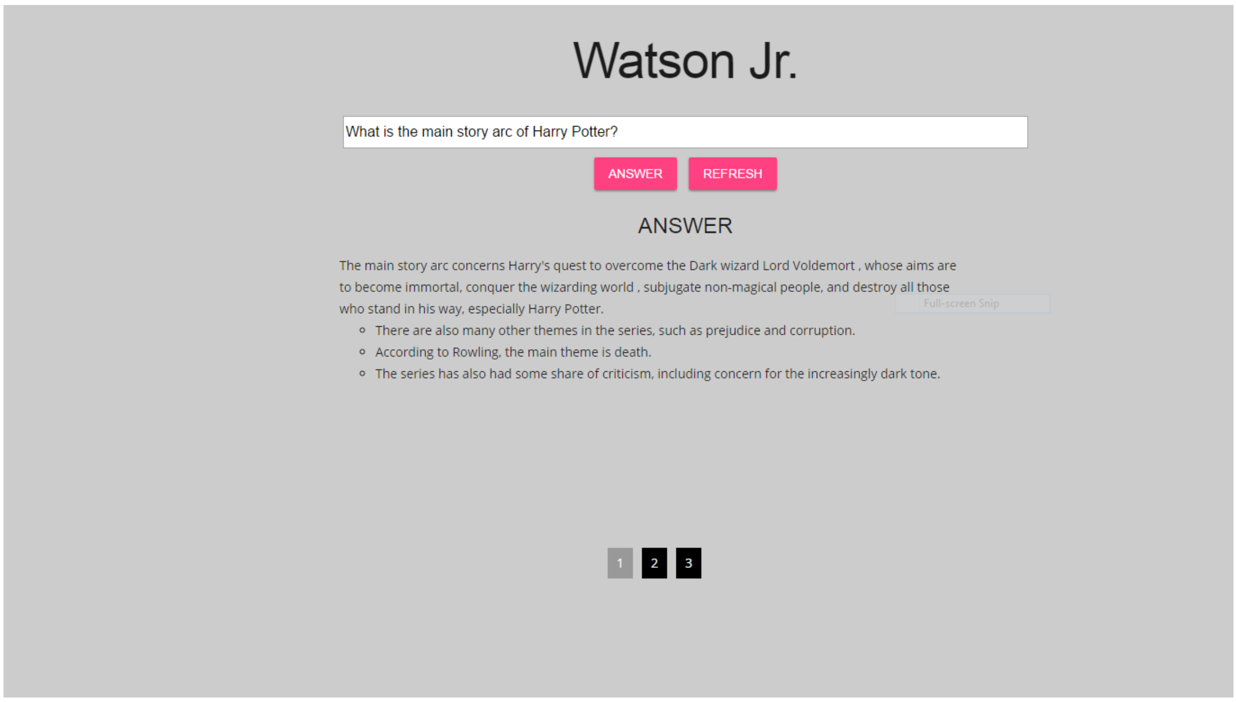


	QuestionType	dev_MAP	dev_MRR	number_of_questions	test_MAP	test_MRR	train_MAP	train_MRR
0	DESC	0.585377	0.584083	378	0.639069	0.659858	0.634874	0.659581
1	NUM	0.650050	0.670290	259	0.641807	0.648404	0.667713	0.669459
2	ENTY	0.729167	0.729167	183	0.562273	0.568437	0.670528	0.687231
3	LOC	0.769826	0.766667	207	0.732415	0.750510	0.733390	0.745532
4	HUM	0.655580	0.655580	212	0.713981	0.713400	0.623217	0.635530

Figure 4.8.2: Performance for various Question types

4.9 End to End system

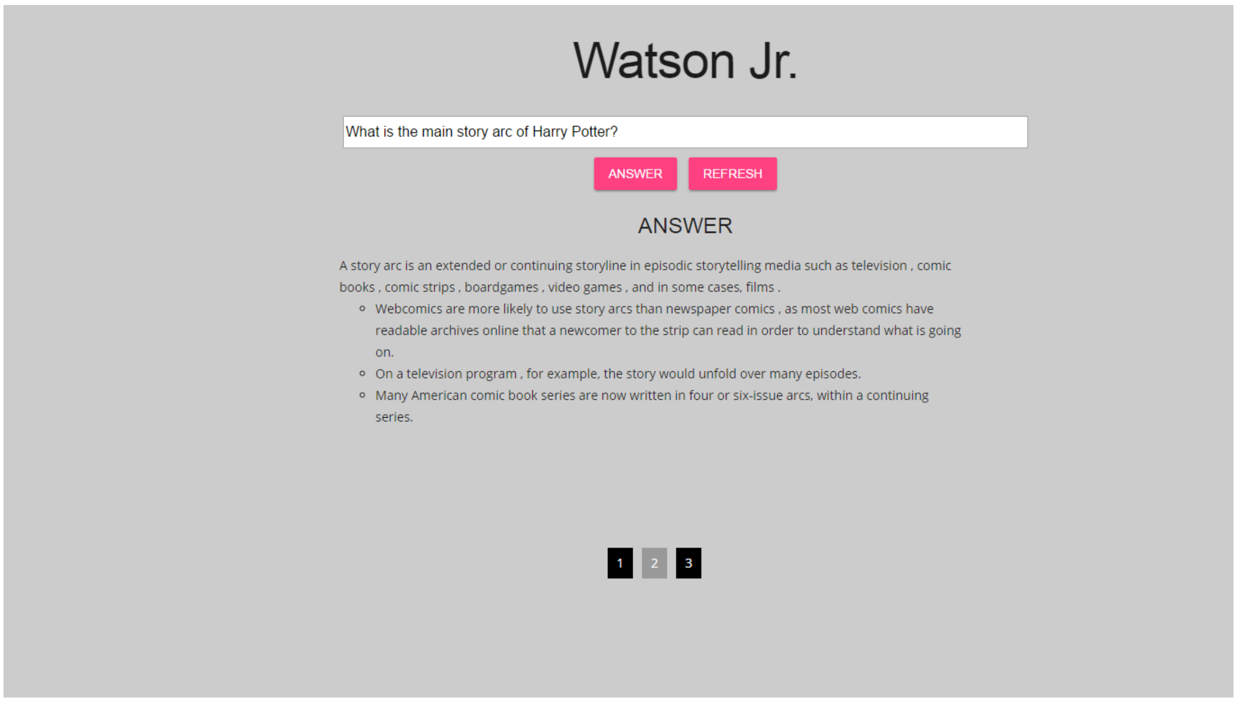
The system retrieves a ranked list of 3 answers per question. The three answers are displayed as a slider on the front end. Each of the answers have a prediction list of answer sentences for the user's next query. There are 3 such sentences and they can be seen as an unordered list under each of the answer sentences. The Fig 4.9.1., 4.9.2., 4.9.3., show the 3 answer sentences and the corresponding prediction sentences for a sample user query. It can be seen that the right answer for the question is the first answer retrieved by the system. The other two answers try to answer the DESC question that was posed using the keywords story arc. The prediction sentences answer the probable next user query.



The main story arc concerns Harry's quest to overcome the Dark wizard Lord Voldemort , whose aims are to become immortal, conquer the wizarding world , subjugate non-magical people, and destroy all those who stand in his way, especially Harry Potter.

- There are also many other themes in the series, such as prejudice and corruption.
- According to Rowling, the main theme is death.
- The series has also had some share of criticism, including concern for the increasingly dark tone.

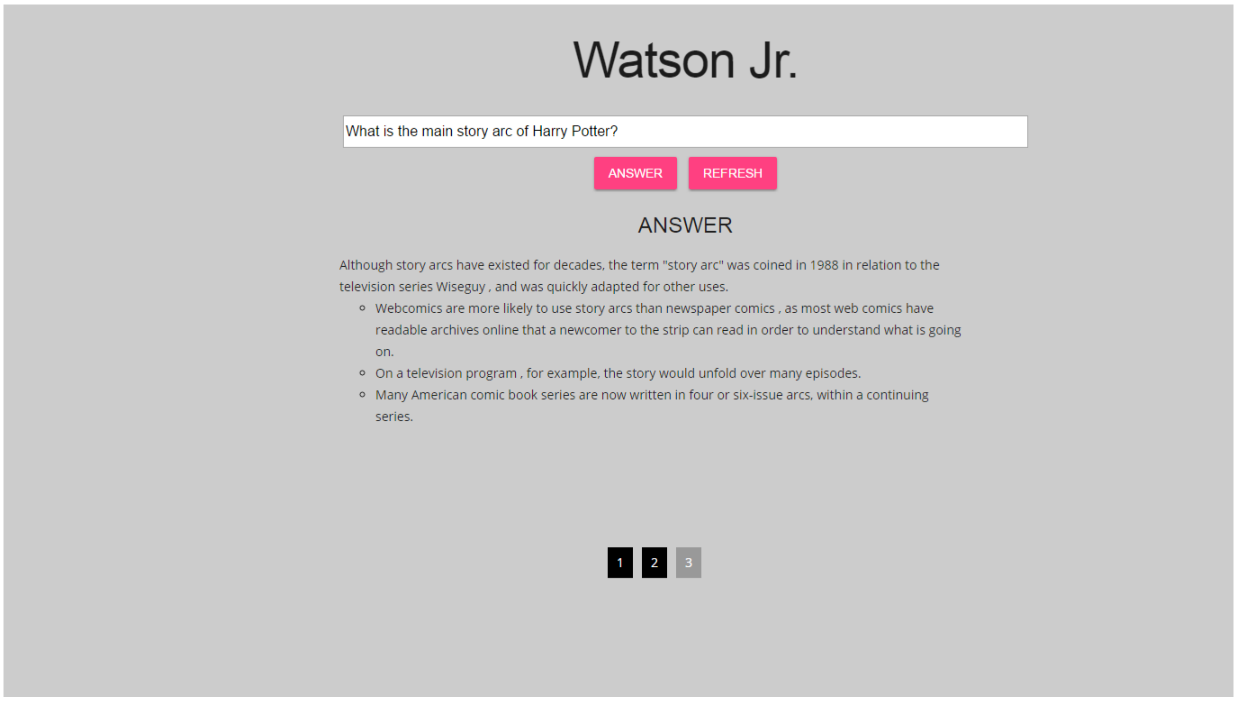
Figure 4.9.1: Rank 1 answer with predictions under it



A story arc is an extended or continuing storyline in episodic storytelling media such as television , comic books , comic strips , boardgames , video games , and in some cases, films .

- Webcomics are more likely to use story arcs than newspaper comics , as most web comics have readable archives online that a newcomer to the strip can read in order to understand what is going on.
- On a television program , for example, the story would unfold over many episodes.
- Many American comic book series are now written in four or six-issue arcs, within a continuing series.

Figure 4.9.2: Rank 2 answer with predictions under it



What is the main story arc of Harry Potter?

Although story arcs have existed for decades, the term "story arc" was coined in 1988 in relation to the television series Wiseguy , and was quickly adapted for other uses.

- Webcomics are more likely to use story arcs than newspaper comics , as most web comics have readable archives online that a newcomer to the strip can read in order to understand what is going on.
- On a television program , for example, the story would unfold over many episodes.
- Many American comic book series are now written in four or six-issue arcs, within a continuing series.

Figure 4.9.3: Rank 3 answer with predictions under it

Chapter 5

Conclusion and Future Work

The system is successful in answering simple factoid questions by retrieving answer sentences from a collection of documents with a high degree of accuracy. The proposed system performs better than any other system currently present in the field. The current best system has an MAP and MRR score of 0.65 and 0.665 respectively. Our model outperforms this by getting MAP and MRR scores of 0.68 and 0.69 respectively. Our system performs extremely well for ENTY and LOC type questions with scores of over 0.7. It underperforms for DESC questions with a score less than 0.6. This can be attributed the wide spectrum of features that have been used in building the binary classifier. The features cover syntactic, semantic and string matching domains. The topic vector innovation helped improve the model performance by a very large amount. The system also introduced a new idea to map the question space to the answer space using average topic vectors.

We also introduced a user question prediction system which has not been explored yet. We make use of a weighted edge graph to store the user interactions with the system. The data stored in this graph is later used for predicting the user's next question and retrieving the possible answers to that question. Also, document clustering ensures that retrieval speed is faster by performing search space reduction. Overall, the proposed system takes us one more step closer to solving the problem of question answering and ensuring good precision, recall and retrieval time.

There is still scope for the model to be improved. Additional features can be explored and more complex machine learning algorithms can be used. Parallel and distributed processing architectures can be experimented to obtain even quicker results. The model can be linked to the internet and allowed to work on a dynamic document store. The model coupled with any search engine would retrieve the answers for a given user question. Since the model is built in a very modular way, it can easily be made to work on the web. Additionally, a more generic model can be built which fully captures the characteristics of DESC type questions.

Bibliography

[1] Yang, Yi, Wen-tau Yih, and Christopher Meek. *"WIKIQA: A Challenge Dataset for Open-Domain Question Answering."* Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. 2015.

[2] Yih, Wen-tau, et al. *"Question answering using enhanced lexical semantic models."* (2013).

[3] Yao, Xuchen, et al. *"Answer Extraction as Sequence Tagging with Tree Edit Distance."* HLT-NAACL. 2013.

[4] Yu, Lei, et al. *"Deep learning for answer sentence selection."* arXiv preprint arXiv:1412.1632 (2014).

[5] Wang, Mengqiu, and Christopher D. Manning. *"Probabilistic tree-edit models with structured latent variables for textual entailment and question answering."* Proceedings of the 23rd International Conference on Computational Linguistics. Association for Computational Linguistics, 2010.

[6] Heilman, Michael, and Noah A. Smith. *"Tree edit models for recognizing textual entailments, paraphrases, and answers to questions."* Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, 2010.

[7] Wang, Mengqiu, Noah A. Smith, and Teruko Mitamura. *"What is the Jeopardy Model? A Quasi-Synchronous Grammar for QA."* EMNLP-CoNLL. Vol. 7. 2007.

[8] Li, Xin, and Dan Roth. *"Learning question classifiers: the role of semantic information."* Natural Language Engineering 12.3 (2006): 229-249.

[9] Yih, Wen-tau, Geoffrey Zweig, and John C. Platt. *"Polarity inducing latent semantic analysis."* Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Association for Computational Linguistics, 2012.

[10] Chang, Ming-Wei, et al. *"Discriminative learning over constrained latent representations."* Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, 2010.

[11] Le, Quoc V., and Tomas Mikolov. *"Distributed representations of sentences and documents."* arXiv preprint arXiv:1405.4053 (2014).

[12] Tomljanovic, J. ; Polytech. of Rijeka, Rijeka, Croatia ; Pavlic, M. ; Katic, M.A.*Intelligent QA Systems: Review of research* , Information and Communication Technology, Electronics and Microelectronics (MIPRO), May 2014

[13] Lorand Dali; Delia Rusu; Bla Fortuna of Joef Stefan Institute*Question Answering Based on Semantic Graphs* , April 2009

[14] Deepak Ravichandran and Eduard Hovy*Learning Surface Text Patterns for a Question Answering System*,In Proceedings of the ACL Conference, 2002

[15] Ehsan Emadzadeh, Azadeh Nikfarjam, Saravanan Muthaiyah, *A Comparative Study on Measure of Semantic Relatedness Function*,vol 1 , 2010 IEEE

[16] Hongming Zhu, Danny Morton, Wenjun Zhou, Qin Liu and You Zhou, *Multi-indexed Graph Based Knowledge Storage System*, WISE 2013 Workshops 2013.

[17] Pasca, M., Harabagiu, S.: High Performance Question/Answering. In: Preceedings of SIGIR 2001, pp. 366-374 (2001).

[18] Dell Zhang and Wee Sun Lee, *A Web-based Question Answering System*, 2002.

[19] Barry Schiffman and Kathleen R. McKeown, *Question Answering using Integrated Information Retrieval and Information Extraction*

[20] Green, B. et al, 1961, *BASEBALL: An automatic question answerer.*, In Proceedings of Western Joint IRE-AIEE-ACM Computing Conference, Los Angeles, p.219-224.

[21] Woods, W., 1973, *Progress in Natural Language Understanding: An Application to Lunar Geology.*, In Proceedings of the National Conference of the American Federation of Information Processing Societies, p.441-450.

[22] B. Katz, *Annotating the World Wide Web using natural language*, Proc. of RIAO 97, MCGill University, Canada, 1997.

[23] Kangavari, M., Ghandchi, S. & Golpour, M., 2008. *Information Retrieval: Improving Question Answering Systems by Query Reformulation and Answer Validation.* World Academy of Science, Engineering and Technology

[24] O. Ferret, B. Grau, M. Hurault-Plantet, G. Illouz, L. Monceaux, I. Robba, and A. Vilnat, *Finding an answer based on the recognition of the question focus*, NIST Special Publication, pages 362-370, 2002.

[25] Christensen, H.U., Ortiz-Arroyo, D. *Analysis and Modeling of Effective Passage Retrieval Mechanisms in QAS*, Electrical and Electronics Engineering, 2006 3rd International Conference, pages 1-4, 2006

[26] Ian Roberts, Robert Gaizauskas *Evaluating Passage Retrieval Approaches for Question Answering*, 26th European Conference on Information Retrieval, pages 72-84, 2003

[27] Xin Li, Enhong Chen *Graph-Based Answer Passage Ranking for Question Answering*, Computational Intelligence and Security (CIS), 2010 International Conference, pages 634 - 638 , 2010

[28] David Ahn and Valentin Jijkoun and Gilad Mishne and Karin Mller and Maarten de Rijke and Stefan Schlobach *Using Wikipedia at the TREC QA Track*, 2004

[29] Silviu Cucerzan and Eugene Agichtein *Factoid Question Answering over Unstructured and Structured Web Content*, 1998

[30] Poonam Gupta and Vishal Gupta *A Survey of Text Question Answering Techniques*, International Journal of Computer Applications ,Vol 53, 2012

[31] Tomljanovic, Jasminka, Mario Pavlic, and Martina Asenbrener Katic. *"Intelligent questionAnswering systems: Review of research."* Information and Communication

Technology, Electronics and Microelectronics (MIPRO), 2014 37th International Convention on. IEEE, 2014.

[32] Dwivedi, Sanjay K., and Vaishali Singh. *"Research and Reviews in Question Answering System."* Procedia Technology 10 (2013): 417-424.

[33] Dragomir R. Radev, Hong Qi, Harris Wu and Weiguo Fan *Evaluating Web-based Question Answering Systems*, Ann Arbor 1001 (2002): 48109

[34] Rajendra Kumar Roul, Omanwar Rohit Devanand and S.K. Sahay, *Web document clustering and ranking using tfidf based apriori approach*, arXiv preprint arXiv:1406.5617 (2014).

[35] Unger, Christina, et al. *Template-based question answering over RDF data.* Proceedings of the 21st international conference on World Wide Web. ACM, 2012.

[36] Lally, Adam, et al. *"Question analysis: How Watson reads a clue."* IBM Journal of Research and Development 56.3.4 (2012): 2-1.

[37] Ferrucci, David, et al. *"Building Watson: An overview of the DeepQA project."* AI magazine 31.3 (2010): 59-79.

[38] Bordes, Antoine, et al. *"Large-scale Simple Question Answering with Memory Networks."* arXiv preprint arXiv:1506.02075 (2015).

[39] Steinbach, Michael, George Karypis, and Vipin Kumar. *"A comparison of document clustering techniques."* KDD workshop on text mining. Vol. 400. No. 1. 2000.

[40] Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. 2014. *Deep learning for answer sentence selection* In Proceedings of the Deep Learning and Representation Learning Workshop: NIPS-2014.

Appendices

Appendix A

Turnitin Originality Report

a

ORIGINALITY REPORT

19%	16%	12%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	en.wikipedia.org Internet Source	4%
2	aclweb.org Internet Source	2%
3	arxiv.org Internet Source	1%
4	citeseerx.ist.psu.edu Internet Source	1%
5	www.nitk.ac.in Internet Source	1%
6	www.cs.cmu.edu Internet Source	1%
7	www.ijeat.org Internet Source	1%
8	www.patinformatics.com Internet Source	1%
9	www.ijecs.in Internet Source	1%
10	Lecture Notes in Computer Science, 2015. Publication	1%

11	www.reddit.com Internet Source	<1%
12	Advances in Intelligent Systems and Computing, 2016. Publication	<1%
13	ieeexplore.ieee.org Internet Source	<1%
14	"Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks", Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR 15, 2015. Publication	<1%
15	Ray, S.K.. "A semantic approach for question classification using WordNet and Wikipedia", Pattern Recognition Letters, 20101001 Publication	<1%
16	Bohac, Marek, Jiri Malek, and Karel Blavka. "Iterative grapheme-to-phoneme alignment for the training of WFST-based phonetic conversion", 2013 36th International Conference on Telecommunications and Signal Processing (TSP), 2013. Publication	<1%
17	www.edureka.co Internet Source	<1%

18	www.chalapathiengg.ac.in Internet Source	<1%
19	Lecture Notes in Computer Science, 2011. Publication	<1%
20	pi7.fernuni-hagen.de Internet Source	<1%
21	Lecture Notes in Computer Science, 2006. Publication	<1%
22	math.stackexchange.com Internet Source	<1%
23	citeseer.ist.psu.edu Internet Source	<1%
24	Li, Xin, and Enhong Chen. "Graph-Based Answer Passage Ranking for Question Answering", 2010 International Conference on Computational Intelligence and Security, 2010. Publication	<1%
25	Daniel Ortiz-Arroyo. "Analysis and Modeling of Effective Passage Retrieval Mechanisms in QAS", 2006 3rd International Conference on Electrical and Electronics Engineering, 09/2006 Publication	<1%
26	Lecture Notes in Computer Science, 2009. Publication	<1%

27	Lecture Notes in Computer Science, 2005. Publication	<1%
28	Lecture Notes in Computer Science, 2014. Publication	<1%
29	Advances in Intelligent Systems and Computing, 2014. Publication	<1%
30	www.cmpe.boun.edu.tr Internet Source	<1%
31	Jihie Kim. "Scaffolding On-Line Discussions with Past Discussions: An Analysis and Pilot Study of PedaBot", Lecture Notes in Computer Science, 2008 Publication	<1%
32	Lecture Notes in Computer Science, 2007. Publication	<1%
33	www.appropedia.org Internet Source	<1%
34	Zhu, Hongming, Danny Morton, Wenjun Zhou, Qin Liu, and You Zhou. "Multi-indexed Graph Based Knowledge Storage System", Lecture Notes in Computer Science, 2014. Publication	<1%
35	docs.di.fc.ul.pt Internet Source	<1%
36	LLORET, ELENA, and MANUEL PALOMAR. "COMPENDIUM: a text summarisation tool	<1%

	for generating summaries of multiple purposes, domains, and genres", Natural Language Engineering, 2013. Publication	
37	is.ijs.si Internet Source	<1%
38	www.ijorcs.org Internet Source	<1%
39	www.sersc.org Internet Source	<1%
40	Lecture Notes in Computer Science, 2016. Publication	<1%
41	TomJohn J.Trevor YehLeeDarrell. "Photo-based question answering", Proceeding of the 16th ACM international conference on Multimedia - MM 08 MM 08, 2008 Publication	<1%
42	eforea.nitk.ac.in Internet Source	<1%
43	Wanpeng Song. "User Feedback for Improving Question Categorization in Web-Based Question Answering Systems", Lecture Notes in Computer Science, 2009 Publication	<1%
44	www.uleth.ca Internet Source	<1%
45	shareok.org	

Internet Source

<1%

46 repository.ubn.ru.nl:8080
Internet Source
<1%

47 Susan Dumais. "Data-driven approaches to information access", Cognitive Science A Multidisciplinary Journal, 5/1/2003
Publication
<1%

48 www.inderscience.com
Internet Source
<1%

49 Grappy, A., B. Grau, M-H. Falco, A-L. Ligozat, I. Robba, and A. Vilnat. "Selecting Answers to Questions from Web Documents by a Robust Validation Process", 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, 2011.
Publication
<1%

50 www.it.iitb.ac.in
Internet Source
<1%

51 Lecture Notes in Computer Science, 2003.
Publication
<1%

52 Nguyen, Thanh, Asim Bhatti, Abbas Khosravi, Sherif Haggag, Douglas Creighton, and Saeid Nahavandi. "Automatic spike sorting by unsupervised clustering with diffusion maps and silhouettes", Neurocomputing, 2015.
Publication
<1%