Question Answer System for Online Feedable New Born Chatbot

Sameera A. Abdul-Kader
School of Computer Science and Electronic Engineering/
University of Essex
Colchester/ UK
University of Diyala/ Diyala/ Iraq
saabdua@essex.ac.uk

Dr. John Woods
School of Computer Science and Electronic Engineering/
University of Essex
Colchester/ UK
woodjt@essex.ac.uk

***Abstract*—Designing a new born Chatbot and feeding it from the web with specific areas of interest is a new research field. Few researchers have investigated empty database Chatbots and populating it from web pages or plain text corpora. Extracting data from web pages needs considerable processing before the response sentences are ready for the Chatbot. Feature extraction is also needed in order to filter and quantify the extracted plain text. In addition, ranking and classification are also required. This paper presents a new method that employs multiple feature extraction methods to quantify text responses for a new born (uneducated) Chatbot. Multiple measurement metrics are examined simultaneously in order to find the nearest match to a query. The nearest matches with the highest score have been obtained by re-ranking the scores of extracted features for text responses. The results show that the highest scored sentences have subjectively a good match to the query. The evaluation results indicate that the performance of the system increases significantly by using cosine similarity to find lexical match between the query and the response sentence rather than Jaccard's coefficient.**

Keywords—New born chatbot; multiple feature extraction; NLP; text quantification; information retrieva; question and answer systems

I. INTRODUCTION

Populating a new born Chatbot database from the web is considered as a new research area. Few researchers have examined educating a new Chatbot in order to incorporate an artificial character. Some authors start database population from web pages or plain text based upon a certain genre or person [1], [2]. Using data from web pages needs numerous operations such as, pre-processing, filtering, mining, and quantification before classification and rank ordering.

Feature extraction methods are used after text mining for both query and response sentences. Quantification uses content analysis involving occurrences, tabulation and statistical semantics for content units [3]. Particular features are used as score measurements for quantification according to statistical calculations.

The aim of this paper is to present a new method that employs various feature extraction methods to quantify text responses for Chatbot queries. The well-known footballer David Beckham is considered as the 'personality' for this Chatbot experiment. The results show that the highest scored sentences match well with subjective analysis and the objective evaluation results show that cosine similarity is more accurate than Jaccard's coefficient to find lexical match between the query and the response sentences.

II. RELATED WORK

The authors in [1] describe an idea to identify significant facts in the text representing the life of a historical figure to build a corresponding Chatbot. This Chatbot should be able to learn (supervised learning) from previous experiences in order to act more realistically. The authors provide a generic form of sentence to solve the problem of learning to enable the Chatbot to acquire as much information as possible relating to the personality and life of the person being simulated. The source of information to feed this Chatbot is websites such as: Wikipedia for unstructured data and DBpedia for structured. NLP techniques are used to convert plain text to structured text and then restructure them into a generic form of a sentence [1]. The input is a collection of factual sentences which are transformed to match the generic form. The authors used an open source Chatbot called ChatScript to design the conversation. ChatScript uses only a simple word (lexical) matching technique to search for the answer. In [4] the authors study supervised discriminative models which learn to choose or rank answers by using examples of question, answer pairs. The representation of Question Answer (Q/A) pair is provided via kernel combinations applied to its members. To reduce the load of large amounts of manual annotation, the authors represent Q/A pairs by means of generalization methods, employing the application of structural kernels to syntactic/semantic structures [4]. This proposal uses more than one matching metric in order to find a better match to the query. The same kind of data as in [1] is used in the proposed system. We combine semantic, lexical, and syntactic features extracted from a text as in [5] in order to obtain an optimized match to a query by quantifying the output results [5]. The contributions are presented in this paper: the first contribution is populating an empty knowledgebase of a new born Chatbot by sentences automatically from the web according to user choice for a figure or an object to simulate by the Chatbot. The second contribution is using a new combination of multiple feature extraction methods to find a best response to a Chatbot query.

978-1-5090-6435-9/17/$31.00 ©2017 IEEE

III. Background

A. New Born Chatbot

To give a suitable response answer to keywords or phrases extracted from speech and to keep conversation continuous, it is required to build a conversational system (program) called a Chatbot (Chatter-Bot). Chatbots can aid in human computer interaction and they have the capability to examine and influence user behavior by asking questions and responding to the questions of the user. The Chatbot is a computer program that emulates intelligent conversation. This program's input is natural language text, and the application gives an answer which is the best intelligent response to the input query. This process is repeated as the conversation carries on and the response should be either text or speech [6].

Trying to build a new born Chatbot is a big challenge. The challenge becomes even more difficult when trying to make it learn. The trickiest part is collecting and processing the data that is used to populate the Chatbot database because the only knowledge the Chatbot has access to, is the information it has learnt itself. So, the data fed into the Chatbot should be selected and filtered carefully using statistical and numerical means.

A new born Chatbot can learn general facts but is often focused towards a specific figure or situation and its database can be updated from the web according to a user request (i.e., the user can choose the figure or the situation they need).

The new born Chatbot has an empty knowledgebase and is considered to be born when its database begins population from the web according to the user's choice of figure or object.

B. Query

The processed data is tested using a sentence or a question that is associated with the figure simulated. This sentence or question is called the query. The query sentence, in this work, is used to test the relevance of the output data in order to measure the performance of the system. The query and input data are analyzed to form their basic components and then the features, that are the measurement metrics, are extracted to quantify the output data. The query here in the experimental part is a set of TREC8 type factual questions about the personal and the career lives of the footballer David Beckham as in the examples below:

1) Who is David Beckham?

2) Who is David Beckham's wife?

3) Which club did David Beckham play for?

4) Where was David Beckham born?

C. Text Quantification

Quantification in the past was the result of counting the frequencies of occurrence and it was a major activity used in content analysis [3]. Quantification of information retrieval means measuring the relationship between a query and a piece of data or text. The relationship can be measured using metrics that are extracted from the query and the text training data. Quantification metrics are chosen depending on the output desired in the experiment. These metrics are called text features that are extracted from the query and the text in order to find common features which helps to quantify similarity or difference between them.

IV. Feature Extraction

In order to be able to quantify the training text according to the query, particular features must be extracted from both the query and the sentence. Then the comparison can be performed between these features. Feature extraction enables to discriminate between text classes [7] and it is considered as an important step to improve the performance of the designed system [8]. A combination of feature extraction methods used together in one system is called a hybrid feature system.

A. Feature selection

The examination of features to be extracted is of primary concern. Arbitrarily selection of features will decrease the accuracy of quantification and thus affects the performance of the system. Relevant features should be selected in order to reduce general data, save storage space in memory, improve the system performance, and to simplify the extracted data. In this paper the features have been selected based on previous experimental work as in [5] which selects lexical, syntactic, and semantic features to quantify the similarity between two sentences.

B. N-Match (Lexical Match)

Lexical analysis means converting a text into a sequence of strings (words) called tokens. Each token has an identified meaning. In this method the number of matching words between the query and the response sentence is detected. The overlapping words can be counted up to the number of words in the query and if the number of matching words is equivalent to the number of words, there is a 100% match.

1) Similarity measurement methods

The association between two words, phrases, or sentences is determined using different comparison metrics such as: similarity, dissimilarity, and distance. As the issue in this paper is matching, then similarity metrics are of this paper's interest. There are a number of lexical similarity measurements. In this paper Jaccard's coefficient and Cosine similarity methods are used to measure lexical similarity between the query sentence and each response sentence.

2) Jaccard's Coefficient

Jaccard's coefficient measures the similarity between two data sets by dividing the number of common properties between the compared sets by total number of features [9]. For example, if X and Y are two sets, then Jaccard's coefficient between them is:

$$J(X, Y) = \frac{|X \cap Y|}{|X \cup Y|} \qquad (1)$$

The technique used in this method is n-grams which is available within the Python library. N-grams is a module that counts the number of words in a sentence or a text and considers each word as a gram. There is a feature in the N-grams module which allows finding the intersectional words between two lists of words. This feature has been employed in this paper to find overlapping words between the query and the response sentence according to the following relation:

$$m1 = r_N \cap q_n \tag{2}$$

And the unity words can alternatively be obtained by using the following relation:

$$m2 = r_N \cup q_n \tag{3}$$

Where,

m1 is the intersectional words between the query and the response sentences.

m2 is the unity words between the query and the response sentences.

r_N is response sentence.

q_n is query sentence.

N is the number of words in response sentence.

n is the number of words in query sentence.

The length of the resultant intersectional list m1 indicates the number of overlapping words (i.e., the number of matches) and the rate of match according to Jaccard's coefficient comes from dividing the overlapping words number by unity words number:

$$L1 = \text{Len}(m1)$$

$$L2 = \text{Len}(m2)$$

$$M(q, r) = \frac{L1}{L2} \tag{4}$$

Where,

M is the rate of lexical match.

L1 is the number of intersectional words.

L2 is the number of unity words.

3) Cosine Similarity

One of the popular similarity metrics used in text document processing techniques is cosine similarity. Cosine similarity is used when the text documents are considered as vectors. The similarity between two documents are here considered as the correlation of two vectors [10]. For example, if two document vectors t_i and t_j are given, Cosine similarity between them is:

$$\text{SIM}_{\cos}(\vec{t_i}, \vec{t_j}) = \frac{\vec{t_i} \cdot \vec{t_j}}{\vec{t_i} \times \vec{t_j}} \tag{5}$$

Where, t_i and t_j are m dimensional vectors of the term set $T = \{t_1, \ldots, m\}$. Each term with its weight is represented by a non-negative dimension. The cosine similarity is bounded between 0 and 1 and it is document length independent.

To apply cosine similarity on the query and the response sentences, the relation will be as follows:

$$\text{SIM}_{\cos}(\vec{q_n}, \vec{r_N}) = \frac{\sum_{i=1}^{i=n} \sum_{j=1}^{j=N} q_n \cdot r_N}{\sqrt{\sum_{i=1}^{n} q_n^2}\sqrt{\sum_{j=1}^{N} r_N^2}} \tag{6}$$

Where, $n = N$

In the implementation part, either of the lexical similarity metrics (Jaccard's coefficient or Cosine similarity) is used at a time to find lexical similarity between the query and the response sentences.

C. NLP match (Syntactic)

Syntactic or Postage analysis is a technique used to analyze a sentence into chunks or phrases depending on Part of Speech Tags (POS) performed according to language and grammar rules. Tokenizing the sentence into separate words is required before applying POS tagging. Syntactic predicate match can be used to obtain the extent of concordance between two compared sentences. The matching can be obtained by comparing the POS tags of the query and the response answer. The function of comparison is shown in the relation below:

$$P = \sum_{i=0}^{i=n} \sum_{j=0}^{j=N} PT \quad \text{where } PT = \begin{cases} 1 & \text{when } POS(i) = POS(j) \\ 0 & \text{when } POS(i) \neq POS(j) \end{cases} \tag{7}$$

Where,

P is the number of pos-tag matches.

PT is the pos-tag.

i refers to position of a pos-tag in the query words.

j refers to position of a pos-tag in the response sentence words.

by dividing by the total number of pos-tags a syntactic or pos-tag match is obtained as shown below:

$$\text{PTM} = \frac{P}{LEN(PT_q \cup PT_r)} \tag{8}$$

Where,

PT_q is pos-tag list of the query sentence.

PT_r is pos-tag list of the response sentence.

LEN is the number of unity postages in the query and the response sentences.

D. Semantic Similarity

Semantic relationship is one of the ways to measure similarity between two sentences. It is used in NLP to compare units of language such as: words, sentences, paragraphs, and documents. "Semantic measures are mathematical tools used to estimate quantitatively and qualitatively the strength of the semantic relationship between units of language, concepts, or instances" using symbolic or numerical description gained according to formal or implicit information comparison for the meaning of these units [11].

In this paper, the entity type semantic method is used (as shown in the equation below) in order to examine the similarity between the two sentences; query and Chatbot response.

$$\text{ENE} = \sum_{k=0}^{k=K} \sum_{l=0}^{l=L} NE \quad \text{where } NE = \begin{cases} 1 & \text{when } k = l \\ 0 & \text{when } k \neq l \end{cases} \tag{9}$$

Where,

ENE is the number of named entity match between the query and the response.

NE is a named entity match.

K indicates the number of named entities in the query sentence.

L indicates the number of named entities in the response sentence.

k indicates a named entity in the query sentence.

l indicates a named entity in the response sentence.

Then the named entity match rate *NEM* is:

$$\text{NEM} = \frac{\text{ENE}}{LEN(NE_q \cup NE_r)} \tag{10}$$

Where,

NE_q is named entity list of the query.

NE_r is named entity list of the response sentence.

LEN is the number of unity words between NE_q and NE_r.

A. Maximum Percentage of match

Total match is calculated by adding the lexical, syntactic, and semantic together in the combination using Jaccard's coefficient as shown below:

$$M_{total} = M(q,r) + PTM(PT) + NEM(NE) \tag{11}$$

In the combination using cosine similarity, the total match equation is:

$$M_{total} = SIM_{cos}(\overrightarrow{q_n}, \overrightarrow{r_N}) + PTM(PT) + NEM(NE) \tag{12}$$

The percentage match is then obtained as in (13) below:

$$Percentage\ match = M_{total} \times 100 \tag{13}$$

The maximum match percentage is obtained by re-ranking the percentage matches calculated above and the higher score can indicate the best match according to the assumptions considered for the output of the proposed.

V. The Proposed System

The proposed system begins with a web crawler with the ability to get plain text from the web using a desired URL as the start/seed. In order to avoid storage limit problems, buffering has been used. The buffer enables the crawler to keep the number of pages within the memory limitation by controlling the generation of new pages. The plain text is pre-processed in order to eliminate unwanted symbols such as punctuations, stop words, or non-English letters and words. After pre-processing the text is mined to give split sentences for the entire text. By using NLTK (Natural Language ToolKit) the sentences are split into individual words and then pos-tagged into speech parts. Then, different feature extraction methods are used to categorize the sentences in the text with regard to a user query and put them into the database. The sentences are rank ordered after quantification according to the features extracted. Referring to the rank order, the best responses for the query can be chosen. The block diagram of the proposed system is shown in Fig. 1.

The proposed web crawler starts with a seed URL for a desired topic or genre in order to produce a new class of text. The seed URL is used to send a request to the associated web page and then to receive an HTML document with the page data. URLs are extracted and saved in a (To Visit) file by parsing the HTML document. Try and Catch tracks the saved URLs in order to check the existence of each of them and then to visit the related new web pages. Plain text is extracted from the web pages of existing URLs and saved into a text file to be processed in the next stage. The process continues up to the last URL in the (To Visit) file. The diagram in Fig. 2 shows the flow of the implemented system.

The text retrieved from the web is read from the (To Visit) file. The plain text may contain different undesired code after acquiring from the HTML. One example is u appearing before each word in the text and this is called UNICODE. So, the text is encoded to ASCII in order to make it easier to deal with the text. The text then is split into sentences using the NLTK sentence tokenizer. The resulting sentences are filtered after that to remove redundant English symbols, punctuations, and non-English letters and symbols. The filtered sentences are broken down into words by the word tokenizing NLTK process and each word is tagged by a part of speech label (POS). NLTK-NE is then used to identify entity names for both the query and the extracted sentences. Features are extracted and compared to calculate the matching score.

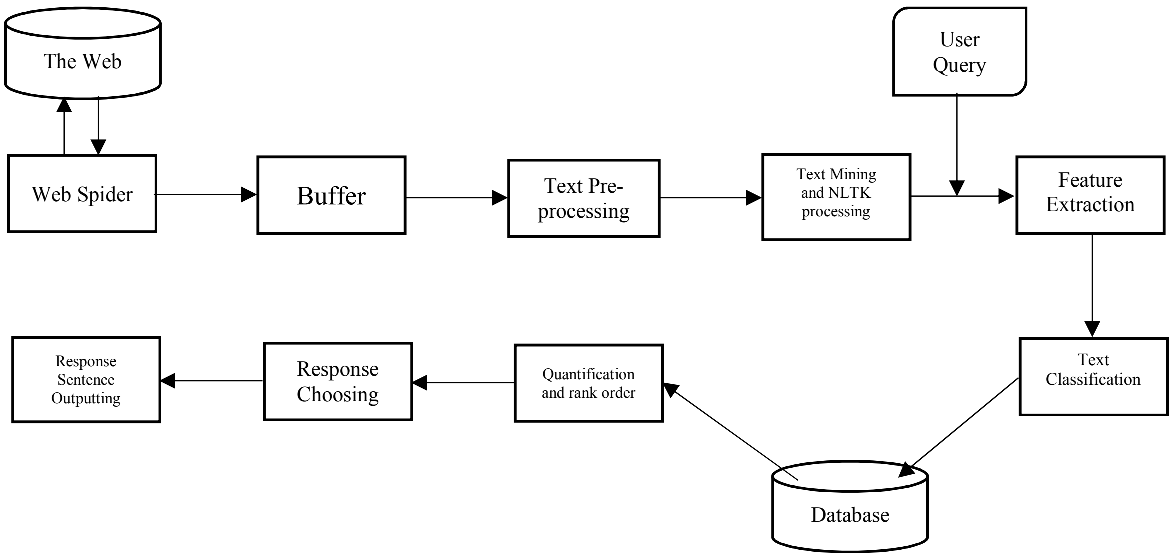

Fig. 1. A block diagram for the proposed system.

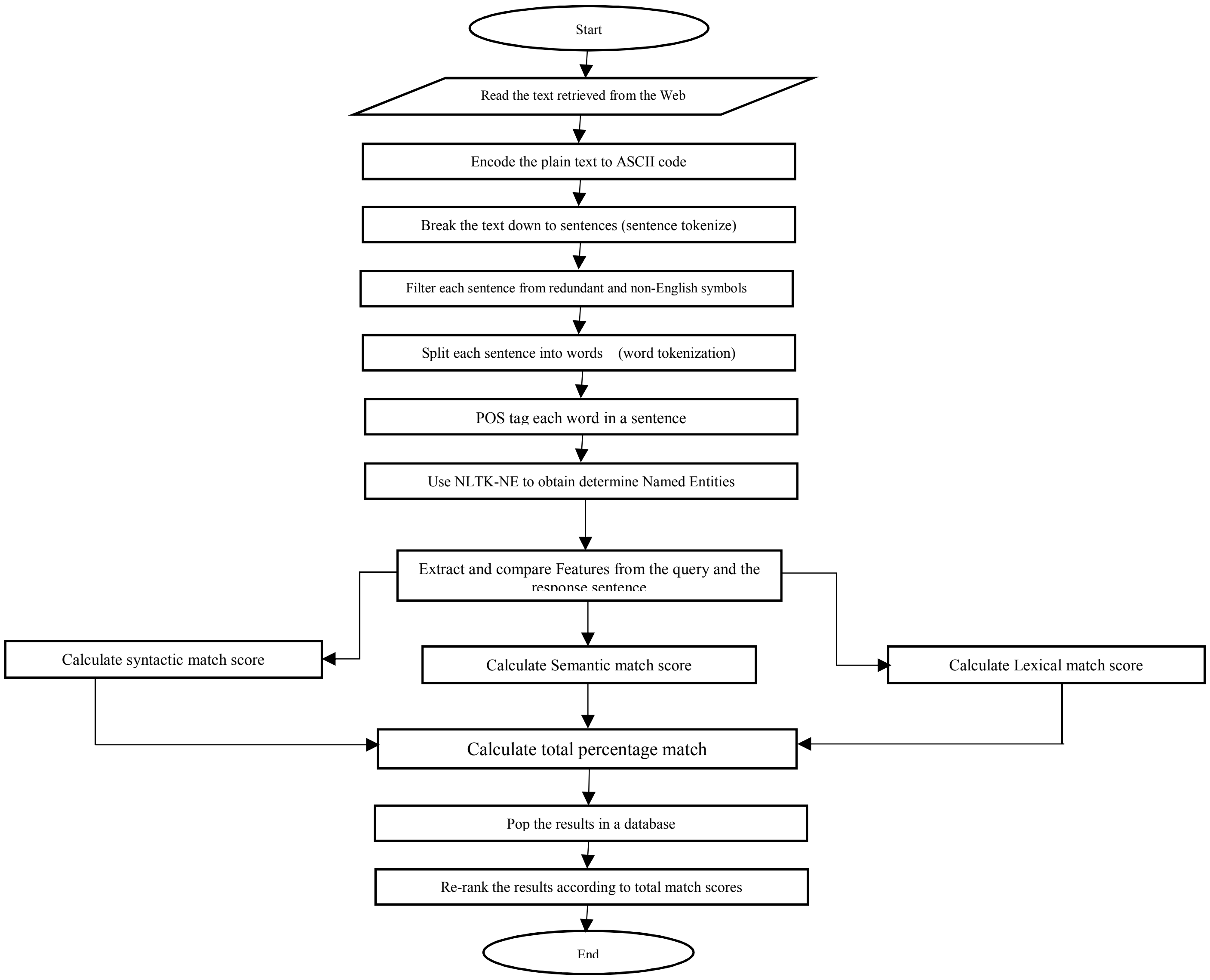
Start
Read the text retrieved from the Web
Encode the plain text to ASCII code
Break the text down to sentences (sentence tokenize)
Filter each sentence from redundant and non-English symbols
Split each sentence into words (word tokenization)
POS tag each word in a sentence
Use NLTK-NE to obtain determine Named Entities
Extract and compare Features from the query and the response sentence
Calculate syntactic match score
Calculate Semantic match score
Calculate Lexical match score
Calculate total percentage match
Pop the results in a database
Re-rank the results according to total match scores
End

Fig. 2. A flowchart diagram of the implemented system.

The scores of syntactic, semantic, and lexical match are calculated then summed to obtain total match score then the percentage match score. The sentences and the calculation results are all placed in an SQLite database prepared for that purpose. The sentences are then re-ranked descending according to their matching scores to evaluate relevance of the highest scores.

The system is implemented in Python and the modules used in the implemented program are: NLTK, re (regular expressions), N-gram, urllib (for web URLs), sqlite3, in addition to BeautifulSoup.

VI. Experimental Results and Discussion

The input data to the system is the unstructured text retrieved from David Beckham's page on Wikipedia (https://en.wikipedia.org/wiki/David_Beckham). Over a hundred URLs associated with Beckham's page on Wikipedia have been accessed to retrieve their unstructured data in order to extract the plain text needed. The output is a set of rank ordered sentences according to a query given after filtering and structuring the unstructured plain text. The resultant output is put into a table of an SQLite database for evaluation purposes. General and personal queries are used to verify the proposed system operation. The resultant sentences have been filtered to get typical length (i.e., not too long) sentences of no more than 21 words in order to solicit clear and concise answers. Examples of the queries, the closest match sentences, and the closest match scores for both combinations in (11) which is the combination including Jaccard's coefficient and (12), the combination including cosine similarity, are tabulated in Table 1. The experimental results are demonstrated in Table 1 give the highest scored and the closest matches out of over 2000 records.

TABLE I. Implementation Results

No.	**Query**	**Nearest match Sentence**	**Combination including Jaccard's coefficient combination Matching Score %**	**Record order in Jaccard's combinatio n**	**Combination including cosine similarity combination Matching Score %**	**Record order in cosine similarity combinatio n**	**No. of records**
1.	Where was David Robert Joseph Beckham born?	Appearances goals David Robert Joseph Beckham OBE b k m/ born May is an English former professional footballer.	37.5	2	42.37	1	2273
2.	Does David Beckham and Victoria Beckham have Children?	Byrne Alla and Perry Simon Victoria and David Beckham Welcome a Daughter.	33.8	2	39.35	1	2168
3.	Did David Beckham play for Manchester United?	At age Beckham s professional career began with Manchester United.	37.85	3	41.48	3	2270

The number of records means the total number of sentences in the database table which the highest and the lowest score sentences is part of. The sentences with 0 scores have been excluded and are not recorded in the database.

VII. Evaluation

The results are evaluated using Mean Reciprocal Rank (MRR) method which is more suitable to measure the performance of the system implemented. MRR is calculated relating to the following relation:

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

Where,

MRR is Mean Reciprocal Rank.

n is number of questions.

i is individual question number.

r_i the reciprocal rank of the correct answer.

Number of questions used to test the system is 28 questions for both Cosine similarity and Jaccard's combinations divided into personal and career groups.

The evaluation results using MRR are illustrated in Table 2.

The examination of Table 2 shows that the performance of the proposed system increases significantly by using Cosine similarity combination shown in (12). The MRR of Cosine similarity combination (52.37) gives improvement by approximately 9 points from the average MRR of *YourQA* (43.25) method used in [4] and 15 points from the combination using Jaccard's coefficient investigated in this paper.

TABLE II. Evaluation Results

No.	**Combination**	**MRR score**
1.	Combination using Jaccard's coefficient	37.13
2.	Combination using cosine similarity	52.37
3.	A. Moschitti, and S. Quarteroni [4].	43.25

VIII. Conclusions

In this paper, a new method that employs multiple feature extraction has been presented to quantify text responses for a new born learning Chatbot. More than one measurement metric has been examined at the same time to find the best match to a Chatbot query. Re-ranking the scores of extracted features for text responses gave the most semantically meaningful sentences. The experimental results show that the highest scored sentences are nearest to a query. Evaluation results show that the system performance rises significantly by using cosine similarity metric for lexical match. In our future work, Chatbot training will be improved by dividing the extracted text into genres to classify the queries into more specialties and used to improve matching. The sentences extracted from the plain text can also be used to produce Question Answer (Q/A) pairs using a technique to extract the question from an answer. The produced Q/A pairs can be used as a Chatbot knowledgebase.

References

[1] E. Haller, and T. Rebedea, “Designing a Chat-bot that Simulates an Historical Figure.” pp. 582-589, 2013.

[2] Z. Yan, N. Duan, J. Bao, P. Chen, M. Zhou, Z. Li, and J. Zhou, “DocChat: an information retrieval approach for chatbot engines using unstructured documents.”, 2016.

[3] R. Franzosi, “Content analysis: Objective, systematic, and quantitative description of content,” Content analysis, vol. 1, pp. XXI-XLX, 2008.

[4] A. Moschitti, and S. Quarteroni, “Linguistic kernels for answer re-ranking in question answering systems,” Information Processing & Management, vol. 47, no. 6, pp. 825-842, 2011.

[5] N. Kambhatla, “Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations.” p. 22, 2004.

[6] S. A. Abdul-Kader, and J. Woods, “Survey on Chatbot Design Techniques in Speech Conversation Systems,” IJACSA, vol. 6, no. 7, pp. 72-80, July 2015.

[7] C. Lee, and D. A. Landgrebe, “Feature extraction based on decision boundaries,” Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 15, no. 4, pp. 388-400, 1993.

[8] D. Trier, A. K. Jain, and T. Taxt, “Feature extraction methods for character recognition-a survey,” Pattern recognition, vol. 29, no. 4, pp. 641-662, 1996.

[9] S. Niwattanakul, J. Singthongchai, E. Naenudorn, and S. Wanapu, “Using of Jaccard coefficient for keywords similarity.” pp. 13-15, 2013.

[10] A. Huang, “Similarity measures for text document clustering.” pp. 49-56, 2008.

[11] S. Harispe, S. Ranwez, S. Janaqi, and J. Montmain, “Semantic Measures for the Comparison of Units of Language, Concepts or Instances from Text and Knowledge Base Analysis,” arXiv preprint arXiv:1310.1285, 2013.