ADAPTIVE ARTIFICIAL INTELLIGENT QUESTION ANSWER SYSTEM

17-107

Final Report

(Proposal Documentation submitted in partial fulfilment of the requirement for the Degree of Bachelor of Science Special (Honours) In Information Technology)

Akram M.R. (IT14109386)
Deleepa Perera (IT14103100)
Singhabahu C.P. (IT14126802)
Saad M.S.M. (IT14109072)

Bachelor of Science (Honours) in Information Technology
(Specialization in Software Engineering)

Department of Information Technology

Sri Lanka Institute of Information Technology

March 2017

DECLARATION

We declare that this is our own work and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Name	Student ID	Signature
Akram M.R.	IT14109386	
Deleepa Perera	IT14103100	
Singhabahu C.P.	IT14126802	
Saad M.S.M.	IT14109072	

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

…… ……..

Signature of the supervisor Date

ABSTRACT

This proposal is for the project titled ‘Adaptive Artificial Intelligent Question Answer Platform’. The goal of this project is to build a Question Answer system that is capable of processing the information in a large dataset and allows the user to gain knowledge from this dataset by asking questions in natural language form. The system is capable of understanding this question responds to the user’s query in natural language form as well. The goal is to make the user feel as if they were interacting with a person. In this proposal we do a literature survey to understand the current status of the QA domain and present some popular solutions that are being used currently. We then make the case for a deep learning alternative in this field and explain why that approach would help negate some of the issues being faced by the other systems. The ‘Objectives’ section explains that we will be using a medical emergency corpus (from the NHS) for proof of concept purposes, thus the sample application we are building will be for this will revolve around this use case. Next we detail out how our proposed systems would work in detail and explain each of the four components in the system.

TABLE OF CONTENT

LIST OF FIGURES

1.0 INTRODUCTION

The information growth rate in the world is increasing at a rapid rate. It has become impossible to keep up with the amount of information that is being added to any given domain. A regular human cannot keep up with all new that gets generated. Due to the wide availability of digital input and output devices and the ease of use of these devices people are creating more and more raw data. If we are able to process this data into meaningful information fast, it would be possible for people to become more productive and to get the information they want faster.

A key idea in social sciences is that a rational human being makes the best decision when he/she is able to access and use all the available relevant information. However this is not very practical. The best way to access all the available information for a given domain would be the internet, but a human is not able to go through all this information and understand and process it in a timely manner to make a good decision.

To a certain extent search engines have been able to tackle this problem. Search engines have become the center of the internet because that is what we use as the gateway to the internet. Rarely do we actually type in a specific web address. Instead we would type in a query into a search engine and use the information that is provided by the search engine to access the data that we want. Over the years search engines have become better and are using much more powerful techniques than simple keyword matching and page link ranking. Still search engines only provide us resources through which we have to sort through and find the answers that we need to find. We cannot use them to give us a direct answer.

We think that this situation can be improved much more and by using deep learning techniques we will be able to create a platform that is able to learn from given data set and then produce direct answers that users can rely on. For the platform to be effective it is important that users can interact with it as naturally as possible. So the user must be able to type in a generic question like they would be talking to a person and the platform should produce an answer that is both factually and grammatically correct. This means that we

would not need to have a specific query language or we would not need to structure the corpus (dataset) manually.

In the next section, the literature survey, we have discussed two approaches that researchers have taken in the past to tackle this problem. Namely, by using natural language processing techniques, information retrieval and template based question answering.

1.1 Literature Survey

Machine learning is a field of Artificial Intelligence that has been gaining a lot of prominence in the current era. It is specifically to do with building systems that are able to learn by themselves without having to be programmed. Deep learning is a subset of techniques of machine learning. Deep learning allows multiple processing layers to breakdown the given data into smaller parts and learn the representations of these data [1]. Deep learning is the state of the art in areas such as speech recognition, natural language understanding, visual object recognition, etc. Convolutional neural networks have brought many breakthroughs in areas such as processing images, pictures and speech, whereas Recurrent networks have been extremely successful in areas such as processing text and speech.

QA is a well researched area from the point of NLP (Natural Language Processing) research. QA has mostly been used to develop intricate dialogue systems such as chat-bots and other systems that mimic human interaction [2]. Traditionally most of these systems use the tried methods of parsing, part-of-speech tagging, etc that come from the domain of NLP research. While there is absolutely nothing wrong with these techniques, they do have their limitations. [3] W.A. Woods et al. shows how we can use NLP as a front end for extracting information from a given query and then translate that into a logical query which can then then be converted into a database query language that can be passed into the underlying database management system. In addition to that there needs to be a lexicon that functions as an admissible vocabulary of the knowledge base so that it is possible to filter out unnecessary terminology. The knowledge base is processed to an ontology that breaks it down into classes, relations and functions [4]. Natural Language Database Interfaces (NLDBIS) are database systems that allow users to access stored data using natural language requests. Some popular commercial systems are IBM's LanguageAccess and Q&A from Symantec [5].

Information retrieval (IR) is another technique that has been used to address the problem of QA. With IR systems pay attention to the organisation, representation and storage of information artifacts such that when a user makes a query the system is able to to return a document or a collection of artifacts that relate to the query [6]. Recent advances in OCR and other text scanning techniques have meant that it is possible to retrieve passages of text rather

than entire documents. However IR is still widely seen as from the document retrieval domain rather than from the QA domain.

Template based question answering is another technique that has been used for QA and is currently being used by the START system which has answered over a million questions since 1993 [7]. START uses natural language annotations to match questions to candidate answers. An annotation will have the structure of 'subject-relationship-object' and when a user asks a question, the question will be matched to all the available annotation entries at the word level (using synonyms, IS-A, etc) and the structure level. When a successful match is found, the annotation will point to an information segment which will be returned as the answer. When new information resources are incorporated into the SMART system, the natural language annotations have to be composed manually [8]. START uses Omnibase as the underlying database system to store information and when the annotation match is found, the database query must be used to retrieve the information. While this system has been relatively successful, it requires a lot of preprocessing which must be done manually.

Our literature survey has found that the QA domain has an active community of researchers and many different approaches have been tried to tackle this problem. While the problem of QA is a very old one, the origins of the problem can be traced back as far as the 1960's, using our access to cheaper and better computational power and newer techniques in data processing we believe we can attempt to solve this issue using a different set of tactics. This will be explained in the next section, the research gap.

1.2 Research Gap

As explained in the introduction, there is a wealth of information on the internet. For any given domain we are able to find a huge amount of information. However to use this information effectively, there needs to be a system to process the data and extract out the meaningful information. Further it is important to provide a simple and seamless way of interacting with this data. This has given rise to the field to natural language question answering where a user must be able to ask a question in everyday language and receive a factually correct answer quickly. In the literature survey we discussed three different techniques that have been used to tackle this problem, NLP, Information Retrieval and Template based question answering. All three methods have flaws where either the accuracy is not high enough or it may take a lot of manual processing and so on. This has meant that while this is an important problem domain due to the high costs there haven't been any commercially viable solutions yet.

The solution that we are proposing for this problem domain is one that is based on deep learning. Deep learning can be defined a subset of machine learning techniques that uses non-linear information processing to identify and extract features and patterns in data, classification and transformations. There are three key reasons that deep learning has become so popular in the recent past. First, the hugely increased processing abilities and availability of general purpose GPU's, the vast amount of training data that has become available and the many advances made in the recent past in the field of deep learning that has made the task of training artificial neural networks more efficient [9].

Deep learning makes extensive use of Artificial Neural Networks (ANN) to complete a given task. ANN's are inspired from the neural networks found in the human brain. The brain consists of an intricate network of neuron cells. Researchers have found success in trying to replicate this structure on silicon chip. Each neuron would consist of what is known as an activation function and the neurons would be connected to each other via connections called tensors [10]. The entire ANN would consist of an input layer of neurons, multiple hidden layers and an output layer. Biases are assigned to neurons and weights are assigned to the tensors. These values influence what eventually becomes the output of the ANN. When

training an ANN we will be adjusting these weights and biases to get the desired output for a known input data set and known scenario [11]. When working with a large ANN with several tens of neurons it can become a tedious task to adjust the weights and biases manually. There are algorithms we can use such as backpropagation to help us adjust these values automatically until we get to the desired output.

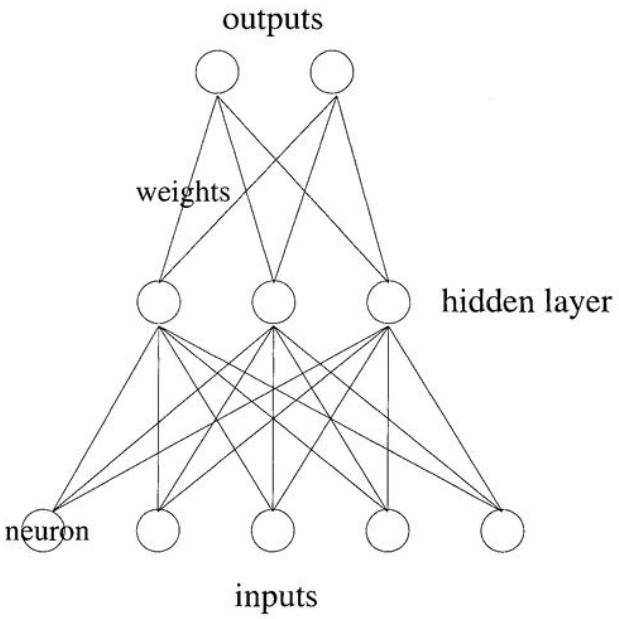

Figure 1.0 Basic representation of an ANN [10]

Since deep neural networks are exemplary at recognising patterns and processing data extremely fast, we believe that by applying deep learning techniques to this problem domain we will be able to overcome many of the drawbacks of the other approaches. We will be able to have a higher accuracy because of the state of the art neural network training paradigms, reduce manual tasks by allowing an ANN to process and structure the corpus and automatically extract the required features and we will not need to use an underlying database engine so therefore we will not need to adopt or develop a different query language.

2.0 OBJECTIVES

2.1 Main Objective

When it comes to deep learning QA systems there are two broad categories that we can target. There are open datasets such as the Allen AI Science and Quiz Bowl datasets, and closed datasets such as the ones provided by Facebook (bAbI) [12]. Open QA systems require using the information provided in the dataset as well as any additional available knowledge. This requires some Information Retrieving techniques. In real world applications this would be the most likely required solution, however for the purposes of this research we have chosen to focus on a closed QA system, where the answer to a question would depend on the given dataset.

To further narrow down our scope we have chosen to build a question answer system for the **medical emergency domain.** As a case study and proof of concept we plan to implement a QA System that focuses only on answering questions related to medical emergency situations and not addressing open domain questions. The scope has been thus narrowed to, first, increase the accuracy of answers provided and second, to ensure that the project can be completed in the allocated time. The corpus will be the ECDS dataset provided by NHS, and the team is currently in the process of negotiating with the responsible people to obtain the dataset.

The end product would be a system that allows the user to ask medical emergency related questions in natural language form and the platform would find the most accurate answer and provide that answer in natural language form as well. The idea is to simulate a situation where the user is interacting with a person in the medical profession as close as possible. The accuracy of the answers will largely depend upon the accuracy of the data in the data set and therefore we cannot guarantee that this will be able to replace an actual medical professional. However the goal in this research is to show that using deep learning techniques we are able to reduce some of the complexities and barriers that are present at the moment and are stopping QA systems from becoming mainstream products. The medical emergency situation was chosen purely out of convenience because of the availability of the dataset. It is only a proof of concept.

The overall system architecture would look something like what's shown below:

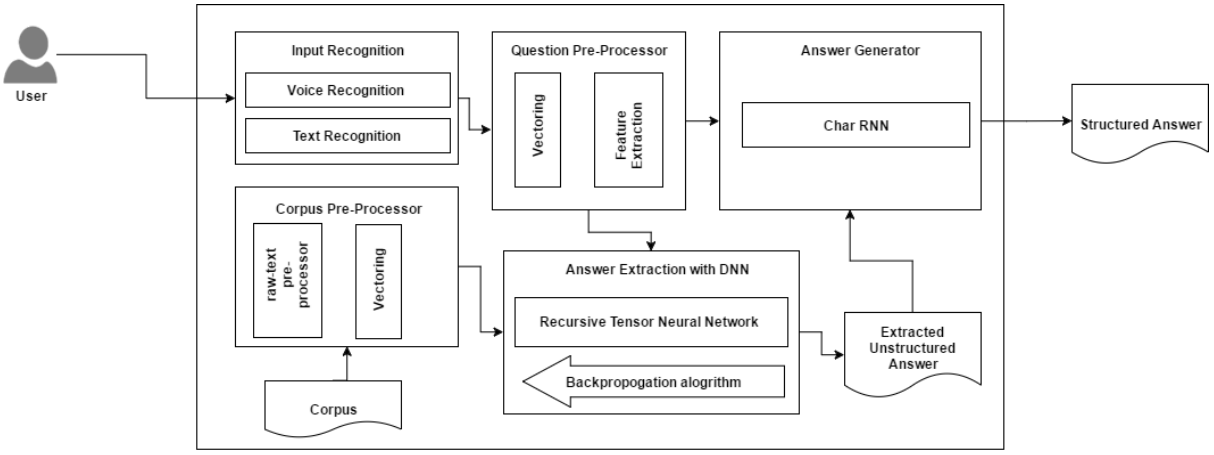

Figure 2.0 System Architecture

In order to achieve the main objective, we have broken it down into four different sub objectives. Each of these sub objectives form a critical part of the system and carry out a critical function. They also have deeply integrated deep learning techniques in each component, which we have described in great detail in the methodology section. In the next section we have a brief overview of what each of the sub objectives are supposed to accomplish.

2.2 Specific Objectives

Aligning a Deep Neural Network based approach for achieving a high accuracy question and answering system is due to its amazing processing and pattern recognition ability.

The traditional approach for Q&A systems is to work with a knowledge source such as an ontology and construct queries by utilizing the question and query the source. This approach has certain downfalls and bottlenecks[13]. Given below are the specific objectives narrowed down to each component in the system.

Question preprocessing

The main objective in this component is to construct a vector space representing the question asked by the user. The vector should be constructed in a way it preserves and maps the semantic relationships of the words in close proximity, in a low dimensional vector space, to

achieve that objective we hope to go ahead with an approach known as word embedding coupled with few other approaches addressing the structure and type of a question.

Deep Neural Network For Answer Extraction

The primary objective of this module is to generate an answer to the pre-processed question by using the structured data available in the pre-processed corpus. The answer is generated and presented only if it exceeds a satisfactory level of confidence in order to ensure the reliability of the system since the proposed system is expected to operate in cases of medical emergencies and the reliability is of paramount importance. The answer generated will be in an abstract form and it will be used by the answer generation block in order to generate a meaningful answer.

Answer Generation

Answer generation is a component that is important to ensure that the user can interact with the platform as naturally as possible. It can be considered as a single-turn dialogue based mechanism for the platform. This means when a user asks a question, it will feel like the user is interacting with a real person. The answer the user gets must be both factually correct and linguistically and grammatically sound. This component will be handling the later part of that.

Corpus Preprocessing

Primary objective of corpus preprocessing is that the preprocessed corpus is the input for training model and the answer is extracted through the training model based on the preprocessed corpus. Therefore the preprocessed corpus should contain a proper representation of data as it is the input for the training model. Intention behind the corpus preprocessing is the transformation of raw text into a representation of vectors in a low dimensional vector space along with maintaining the words which are similar in close proximity through understanding the contextual similarity of words and mapping the semantic relationship of words.

3.0 METHODOLOGY

3.1 Question Preprocessing

In a Q&A based system understanding and processing a question is vital to provide the user with the most appropriate answer. Machines do not understand text as humans do thus questions inputted as texts need to be transformed into a vector that preserves the context of the question that was presented by the user. The objective of this area is to provide the neural network with the most effective vector format that would preserve and best represent the context of a question in a vector so that the DNN can utilize the vector to process and find the most appropriate answer.

For this purpose we will be using few natural language modelling techniques such as word embedding,syntactic analysis, Identification of question type such as a wh-question analysis. Word Embedding is used to map words or phrases from a vocabulary to a corresponding vector. This type of representation has two important and advantageous properties:

1. It is a more efficient representation
2. It is a more expressive representation

Word embedding maps each word to a vector space.The Embedding layer will map each token from the question to its corresponding vector space, which preserves the contextual similarity of words in the vector space.

The embedding layer in the question processing component can be done through a popular pre-trained word embedding model known as word2vec or glove(exact model will be chosen based on further trial and error)[14]. Word2vec is a small two layer neural network. It contains two distinct models (CBOW and skip-gram), each with two different training methods (with/without negative sampling) and other variations [14]. To top that, it also contains a sharp pre-processing pipeline, whose effects on the overall performance is yet to be evaluated.

The other sub-component is Syntactic analysis of a question. Syntactic analysis is the process of identifying the structure of a sentence,The interplay of syntax and semantics of natural language questions is of interest for question representation. Researchers in the area of question understanding reccomend the use of a TreeLSTM neural network for this purpose[15], since it is capable to capture long distance interaction on a tree. The other option would be to go with a chain structured LSTM but the critical downfall of it for this specific task is that it fails to capture long distance interaction on a tree[15].

To obtain the parse tree information some of the available open source parsers such as the Noah's Ark parser [16], or the Standford Core Parser can be used.

Questions by nature are composed to fulfill different types of information. A what question and a how question requires different types of information [15]. In Order to incorporate this a Wh-Analysis will be required, thus an additional layer for question adaption will be required. Wh-word is basically the question word which is one of who, why, where, which, when, how, what and rest. “rest” are the questions that don't have any question word. Example :- Name of a disease that cause bowel bleeding?. This process segregates questions into different types and considers the type of question for answer generation, a recommended approach for this would be to encode question type information into a one hot vector which is a trainable embedding vector [15], and it is incorporated in the training process.

For the purpose of implementation the popular deep neural network library tensor flow along with the python programming language will be used. For the purpose of syntactic analysis the Standford core NLP parser will be used.Python enables us to carry out string manipulation easily unlike other programming languages since user inputs are text based it will be the preferred choice of language.The choice of the above mentioned technology is due to the widely available community support and free distribution of the software.

Work Flow diagram of the question preprocessing component is given below.

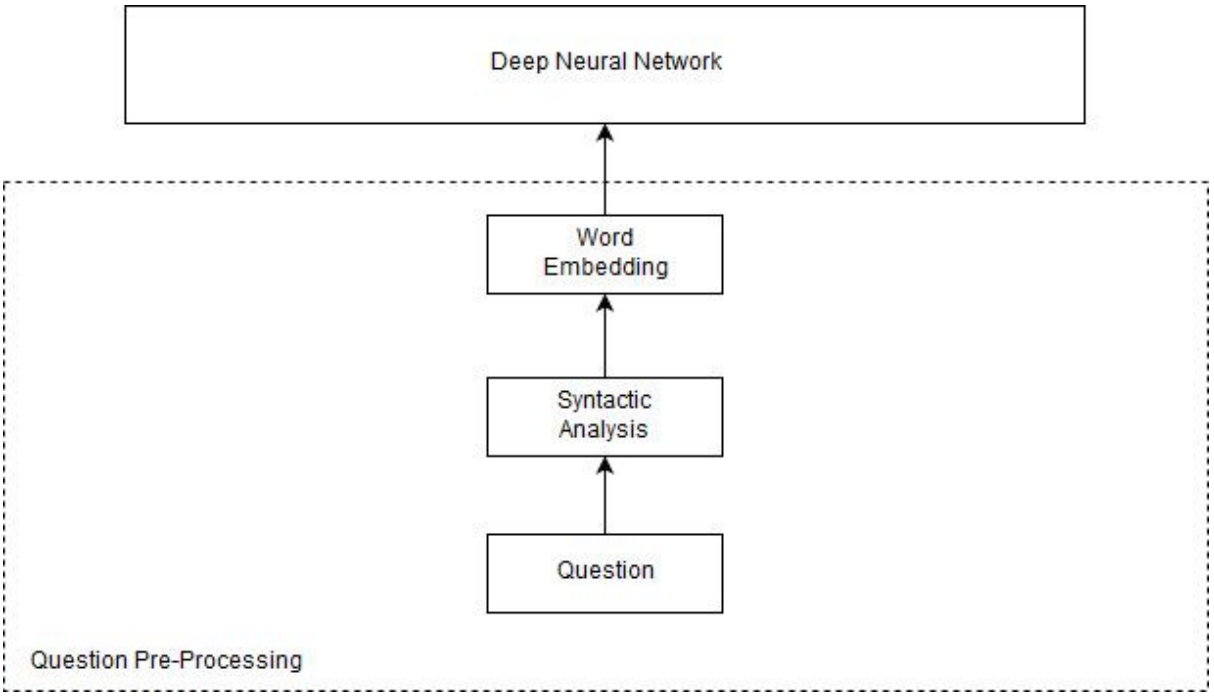

Figure 3.0 Question pre-processing module

3.2 Deep Neural Network For Answer Extraction

As mentioned before, the main objective of this project is to come up with an Artificial Neural Network (ANN) based solution for Intelligent Information Extraction. In here we are particularly focusing on addressing the Information Retrieval in a form intelligent Question and Answering System. In related work, Ontology based information extraction [17], seems to be a recently emerged subfield of information retrieval. However the inability to reason over discrete and their relationships can be identified as a major drawback in these Ontologies and knowledge base information retrieval/extraction systems [18]. Therefore we present a system which is capable of performing this task with the help of ANN particularly Recursive Neural Tensor Networks.

Artificial Neural Networks are known to perform well in intelligent systems when they are properly tweaked and used. However in order to get the optimal performance of such system it is important to select the most appropriate ANN approach and apply it accordingly. After a comprehensive study in current literature on this matter we decided to use Recursive Neural Tensor Network (RNTN) as our main inspiration for realizing this system. The selection of this methodology was done after analyzing the pros and cons as well as the relevance and the suitability of RNTN compared to other subfields in ANN. Some of the popular choices for ANN are Feedforward neural networks, Recurrent neural networks, Neural Tensor networks etc. However in order to understand the reason behind our selection over these approaches, we will first look at the main objectives expected by this module of the system.

This module, namely “Information Extraction Module” basically acts as the part where the actual decision making happens. The primary objective of this module is to generate an answer to the pre-processed question by using the structured data available in the pre-processed corpus. All these inputs will be in a form of vectors. The answer which is the expected output of this module, is generated and presented only if it exceeds a satisfactory level of confidence in order to ensure the reliability of the system since the proposed system is expected to operate in cases of medical emergencies and the reliability is of paramount importance. The answer generated will be in an abstract form and it will be used by the answer generation block in order to generate a meaningful answer.

Having understood the basic objective of the module now let us take a look at the reason behind our proposed methodology in the context of the problem statement. It should be noted that here we are generating answers from a pre-processed corpus which is already available. The Recurrent Neural Networks (RNN) shows promising performance when implemented in system where it is needed to exhibit dynamic temporal behavior in applications like handwriting recognition, speech recognition. However in our case since we are using data from a predefined corpus this might not be the most appropriate solution to address the requirement. On the other hand Neural Tensor Network specifically Recursive Neural Tensor Networks (RNTN) are more appropriate in scenarios where Natural Language Processing (NLP) and Sentiment Analysis are performed. These RNTNs are able to deal with the hierarchical relation in the words of a sentence which will be beneficial in our case.

In order to implement this ANN system, Python along with Tensorflow library will be used as main tools. This choice of the above tools was done considering their performance and the online support available for these tools compared to others. The neural network will be trained with the expectation of performing information extraction on an arbitrary (generic) rather than a specific corpus in order to realize a versatile system. The training dataset will be provided by National Health Services, England. In addition to that possibility of using datasets which are available online will also be explored.

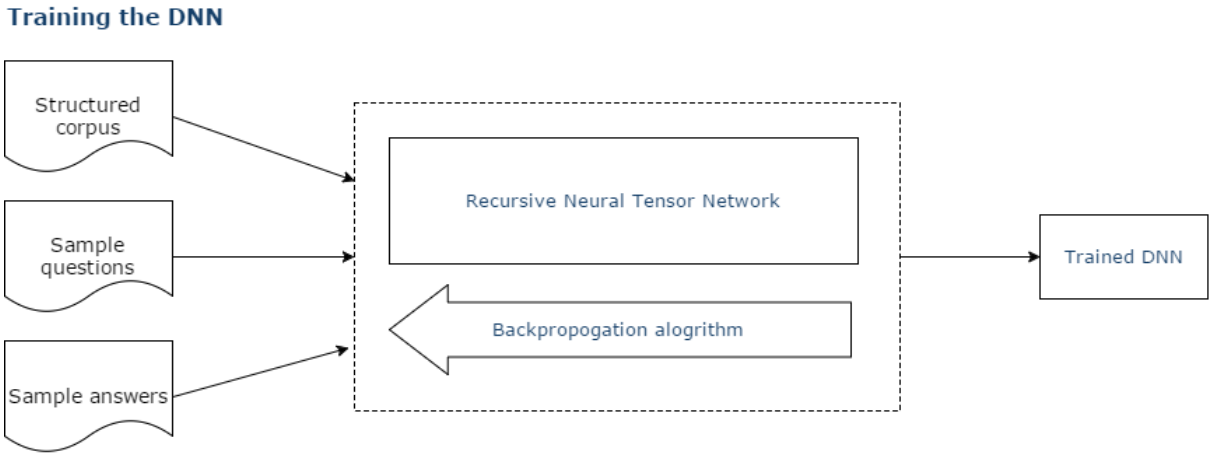

Figure 4.0 Training the RNTN

3.3 Answer Generation

The objective of this component is to generate answers in natural language form after the correct answer has been identified. This component is required because the corpus will not always have the answer in the required format. This means we need to pre-process the answer that we give to the user. Since a goal of this project is to make it extremely user friendly to interact with a large corpus, the answer generation component is necessary to ensure that user is able to interact with the system as naturally as possible.

The challenge with generating natural language from a single piece of text is that it is the reverse approach to what is taken by most other researchers. Usually deep neural networks are used to identify patterns and are used as predictive models. In this case the DNN must be used as a generative model instead. This means given some fragments of a sentence, a well trained model can create a proper sentence in natural language format.

One possible approach is to use a recursive auto encoder model which will be fed with data formatted in a dependency parse tree [19]. In this approach the researchers have constructed a sentence level vector which can be decomposed when required and the same sentence can be retrieved again later on. However by randomising the vector's at the output stage (decomposition) they have managed to generate paraphrased sentence structures from the data that they feed in. This approach requires some complex corpus preprocessing and is therefore a tedious task to carry out.

Another approach to this problem is to use recurrent neural networks (RNN). In a RNN the neurons and tensors connect together to form a directed cycle. This gives the neural network some memory properties. The neurons have an activation time after which they stop firing. This is in contrast to regular feed forward networks. However RNN's have what is often referred to as the 'exploding gradients problem' because of the hidden states that the neurons can go into. This has made training RNN's tough. A recent development in the Hessian-Free optimization technique however has made it effective to train RNN's [20]. Sutskever et al. show how they have trained and optimised a RNN to predict the next character in a stream of

characters [21]. The researchers developed a character level RNN model that was able to generate text with a high level linguistic structure and correct grammatical structure.

The most prominent research that has been done in this field is by a researcher named Andrej Karpathy, a PhD student from Stanford. He created a RNN called Char-RNN that is able to generate new text when a chunk of text is fed. Char-RNN works by analysing the previous characters in a string and guessing what the next characters should be. Therefore, by training a RNN with a large data set such as Project Gutenberg we will be able to train a good RNN model. Then the model can be fed with some primetext which would come from the answer sentence selected in the previous component and the model should be able to generate the required sentence [22].

A RNN acts like a feedback loop where the output of one layer is added to the input and is fed back into the same neural net. This means a RNN can form a time series as shown below. So it is able to process information in a continuous series and this makes it ideal for forecasting information.

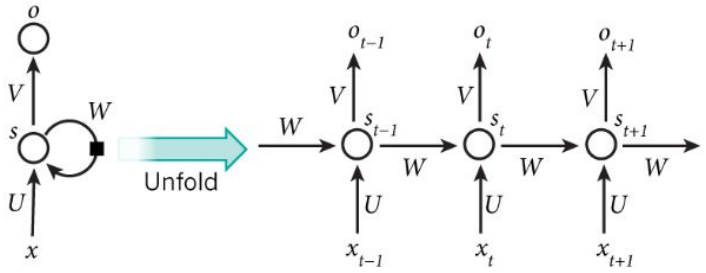

Figure 5.0: A time series RNN [23]

In the diagram above the output W of S_{t-1} becomes the input of the next iteration S_t, along with the new data, U. So the outputs form a time series. We can use this property of RNN's to generate new text from a model that has been trained using a vocabulary that is similar to the knowledge base that we are using. Existing research also shows that this method has proven to be successful were researches like Andrej Karpathy have been able to generate entire Shakespearean paragraphs through the RNN.

The expected outcome of this component would be something like follows:

Question : When was Sri Lanka railways founded?

Identified answer sentence : Sri Lanka railways has not been profitable since it was founded in 1858

Primetext : Sri Lanka railways founded 1858

Expected output : Sri Lanka railways was founded in 1858

3.4 Corpus Preprocessing

Corpus Preprocessing is one of primary component in the research project. Initially the corpus will be unstructured text data, which can be understood by humans, not by machines. It simply implies that, there is a necessary for a transformation of such data because many machine learning algorithms including deep neural networks, require inputs to be vectors of continuous values; they won't just work on plain text or strings.

Since the amount of information is rapidly growing and it is becoming difficult to keep manually create knowledge bases and ontologies uptodate. Than being relying on manually created knowledge bases, applying deep learning techniques to unstructured data or the corpus to identify relationship of words and providing the proper transformation of unstructured data to a representation as an input to the neural network training model, will provide an immense help in extracting the needed answer.

One of the major application of such transformation of data [24], is Word Embedding which is a natural language technique. It is used to map words or phrases from a vocabulary to a corresponding vector of real numbers. Word embedding aims to create a vector representation [25] with a much lower dimensional space. In contrast Bag of Words [26] approach, which often results in huge and sparse vectors. In Bag of Words approach the dimensionality of the vectors representing each document is equal to the size of the vocabulary.

Word Embedding [27] is also used for semantic parsing, to extract the meaning from text to enable Natural Language Understanding. For a language model to understand the meaning of a word, it need to know the contextual similarity of words. For example if we tend to find diseases in sentences, where diabetes, diarrhea, HIV should be of close proximity. So the vectors created by word embedding preserves these similarities along with the words that regularly occur nearby in text will also be in close proximity.

Word embedding is all about building a low dimensional vector representation from corpus, which preserves the contextual similarity of words. There are word embedding techniques such as,

- 1-of-N vec (one-hot-vec)
- GloVe (Global Vectors)
- Word2vec

In a simple 1-of-N [28] encoding transforms categorical features to a format that works better with classification and regression algorithms. In simple terms every element in the vector is associated with a word in the vocabulary. The encoding of a given word is simply the vector in which corresponding element is set to one and all others are set to zero. Suppose we consider a vocabulary of five words; diabetes, diarrhea, HIV, obesity, paralysis. We could encode the the word obesity as [0, 0, 1, 0, 0]. In such a scenario, the only comparison that can be made between vectors is equality testing or it engages all features and tell which is present and which is absent for a particular set of output.

GloVe [29] is a 'count based' model. Which means GloVe learn their vectors through collecting word co-occurrence statistics in a form of word co-occurrence matrix **X**. Each element $\mathbf{X}_{ij}$ of such matrix represents how often word ***i*** appears in context of word ***j*** in a large corpus. The number of "contexts" is of course large, since it is essentially combinatorial in size. Then factoring this matrix to yield a lower-dimensional matrix, where each row now yields a vector representation for each word.
In specific, the creators of GloVe illustrate that the "ratio of the co-occurrence probabilities of two words (rather than their co-occurrence probabilities themselves) is what contains information and look to encode this information as vector differences". But when it comes for computation GloVe will be taking more memory [30], because it precomputes the large word into word co-occurrence *(large word x word co-occurrence)* matrix in memory[31]. Also sometimes there is a restriction on vocabulary since GloVe requires memory quadratic in the number of words: it keeps that sparse matrix of all *word x word co-occurrences* in RAM.

Word2vec [32] is a predictive model which is one of the most popular word embedding model. This simply learns relationship between words without any prior knowledge about the domain. The output are vectors, one vector per word with an exceptional linear relationship. Once a vector model is created out of the corpus, word2vec provides two basic tools namely *analogy* and *distance* to use.

- Distance - tool provide a list of words which are closely related to a particular word from the vector model.
- Analogy - tool is provides the ability to query for textual regularities captured in the vector model.

For example, let us assume that we use word2vec to create a vector model of the words appearing in a corpus of medical domain. If the resulting vector space represents diseases and cause of disease is projected in a two dimensional vector space, we can observe a relationship between each disease and the cause of the disease, and also similar diseases are placed closed to each other in vector space.

Further word2vec is based on two architecture: *continous bags of words (CBOW* and *skip-gram (SG)*. Since word2vec as no built-in functionalities for term normalisation, the corpus needs to processed before they could be used for word2vec. Unprocessed corpus contains syntactic variations, stopwords, punctuations which has a negative impact on on how word2vec indexes the term and which will affect the quality of vector space representation.

Before providing the corpus to the word2vec it needs to be processed as follow,

- All punctuations and unnecessary white spaces needs to be removed.
    - Eg: the term *obesity* and *obesity. (with full stop)* can be indexed as two separate terms.
- Transforming all words to lower-case.
    - Eg: the terms *Obesity obesity OBESITY* can indexed as three separate terms.
- Merging muti-word terms.
    - Eg: the terms *Human Immunodeficiency Virus* transformed to *human_immunodeficiency_virus* so that word2vec can identify it as a single term.

  (Merging multi word terms is not yet finalized, decision will be taken during the implementation phase. To achieve this should create a separate dictionary of multi word terms for the selected domain using knowledge source.)

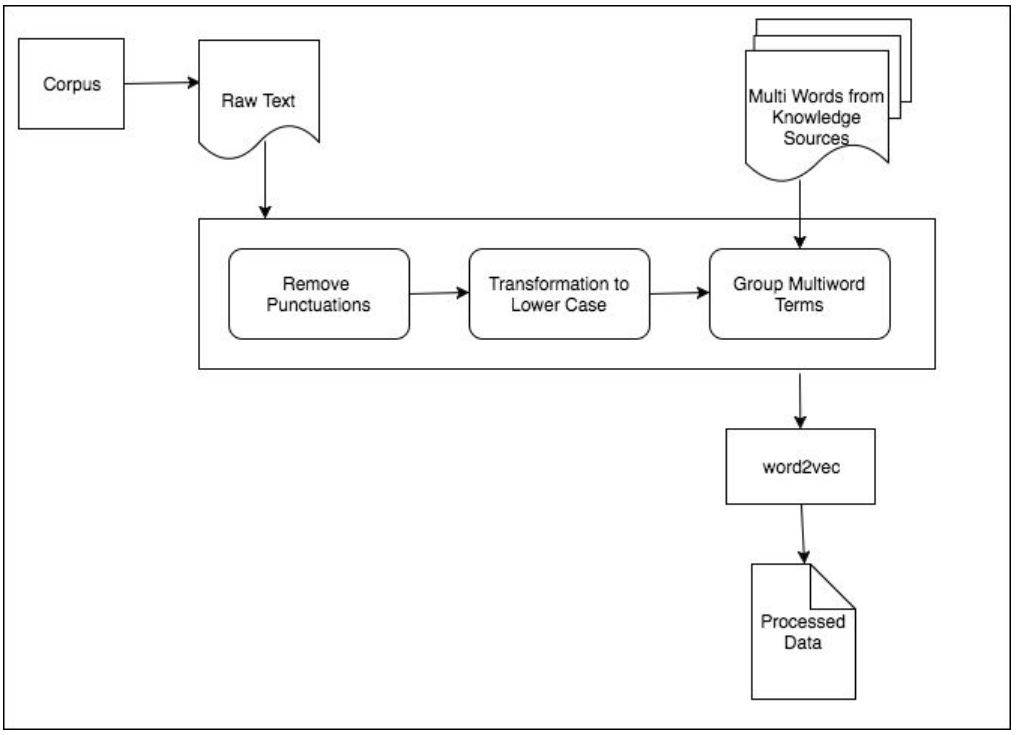

Figure 6.0 Workflow of Corpus Preprocessing

The preferred choice of technology for this component will be Python along with Tensorflow. Tensorflow is a popular machine learning platform with a lot of community support and ease of use, along with Tensorflow keras.io will also be used. Keras is a superset of Tensorflow. Python enables us to carry out string manipulation easily unlike other programming languages since corpuses are text based it will be the preferred choice of language.

4.0 GANTT CHART

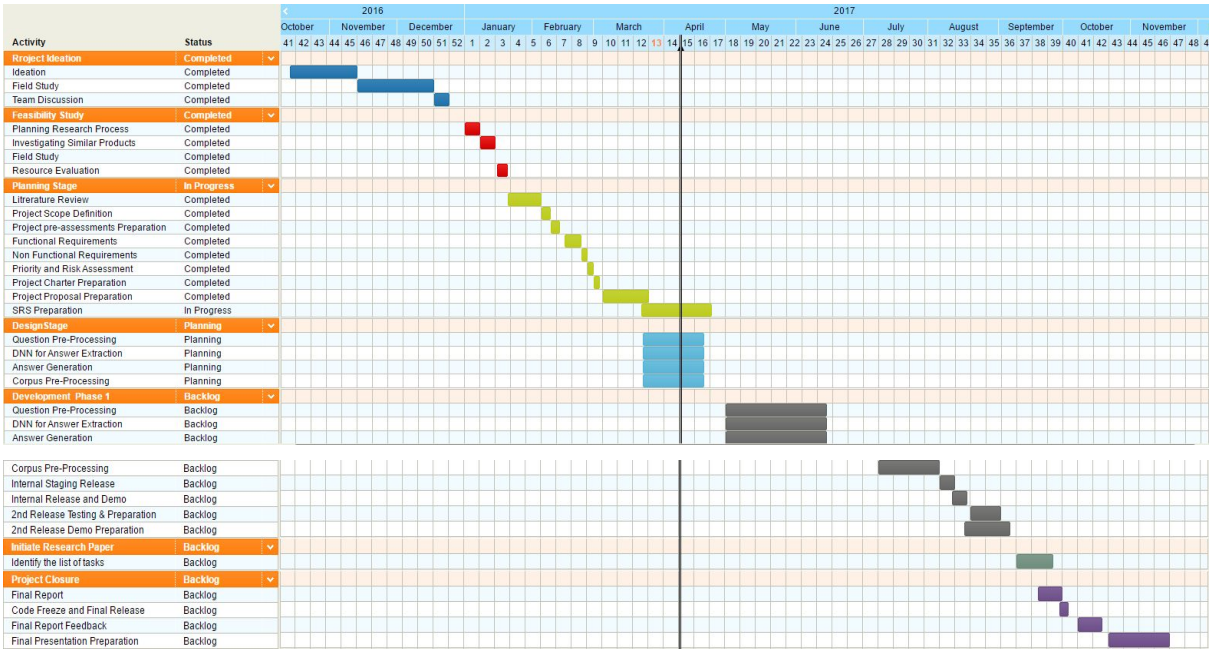

Figure 7.0 Gantt Chart

5.0 WORK BREAKDOWN

Task	Component	Responsible
Requirements Gathering		Rifhan Akram Deleepa Perera Singhabahu C.P. Saad M.S.M.
System Design		Rifhan Akram Deleepa Perera Singhabahu C.P. Saad M.S.M.
Research		Rifhan Akram Deleepa Perera Singhabahu C.P. Saad M.S.M.
Implementation	Question Preprocessing Component	Rifhan Akram
	Deep Neural Network Component	Singhabahu C.P.
	Answer Generation Component	Deleepa Perera
	Corpus Preprocessing Component	Saad M.S.M.
Web Interface		Singhabahu C.P. Saad M.S.M.
Mobile Application		Rifhan Akram

(Android only)		Deleepa Perera
System Testing		Rifhan Akram Deleepa Perera Singhabahu C.P. Saad M.S.M.
Continuous Integration and Deployment		Singhabahu C.P.

6.0 DESCRIPTION OF PERSONAL AND FACILITIES

The description of the personnel involved in this project is as follows:

Supervisor: Mr. Yashas Mallawarachi

External supervisor: Mr. Anupiya Nugaliyada

Implementation team:

Akram M.R. (Leader)

Deleepa Perera

Singhabahu C.P.

Saad M.S.M.

7.0 REFERENCES

[1] Y. LeCun, Y. Bengio and G. Hinton, "Deep learning", Nature, vol. 521, pp. 436 – 444, 2015

[2] Silvia Quarteroni, “A Chatbot-based Interactive Question Answering System”, 11th Workshop on the Semantics and Pragmatics of Dialogue, 2007

[3] W.A Woods, R.M. Kaplan and B. Nash-Webber, “The lunar sciences natural language information system”, BBN Rep. 2378, Bolt Beranek and Newman, Cambridge, Mass., USA, 1977

[4] T.R. Gruber, “A translation approach to portable ontology specifications”, Knowledge Acquisition, 5 (2), 1993.

[5] R. Dale, H. Moisl and H. Sommers, Handbook of Natural Language Processing, 1st ed. New York: Marcel Dekker AG, 2006, pp. 215 - 250.

[6] L. Hirschman and R. Gaizauskas, “Natural language question answering: the view from here”, Natural Language Engineering, 7 (4), 2001, pp. 275-300.

[7] "The START Natural Language Question Answering System", Start.csail.mit.edu, 2017. [Online]. Available: http://start.csail.mit.edu/index.php. [Accessed: 26- Mar- 2017]

[8] B. Katz, G. Borchardt and S. Felshin, “Natural Language Annotations for Question Answering”, Proceedings of the 19th International FLAIRS Conference (FLAIRS 2006), 2006

[9] L. Deng and D. Yu, "Deep Learning: Methods and Applications", Foundations and Trends in Signal Processing, vol. 7, no. 34, pp. 197–387, 2013.

[10] S. Wang, Interdisciplinary computing in Java programming language, 1st ed. Boston, Mass.: Kluwer Academic, 2003.

[11] N. Gupta, "Artificial Neural Network", Network and Complex Systems, vol. 3, no. 1, pp. 24-28, 2013.

[12] E. Stroh and P. Mathur, “Question Answering Using Deep Learning”, Stanford Reports

[13] Burger, John, and Claire Cardie. "Q&A Roadmap Paper Page 1 10/24/03 Issues, Tasks And Program Structures To Roadmap Research In Question & Answering (Q&A)". (2017): n. pag. Print.

[14] Ruder, Sebastian. "On Word Embeddings - Part 3: The Secret Ingredients Of Word2vec". *Sebastian Ruder*. N.p., 2017. Web. 26 Mar. 2017.

[15]J. Zhang, X. Zhu, Q. Chen, L. Dai and H. Jiang, "Exploring Quesiton Understanding and Adaptation in Neural-Network-Based Question Answering", 2017.

[16] "Noahs-ARK/semafor", *GitHub*, 2017. [Online]. Available: https://github.com/Noahs-ARK/semafor. [Accessed: 27- Mar- 2017].

[17] Daya C. Wimalasuriya Dejing Dou, “Ontology-based information extraction: An introduction and a survey of current approaches “, Journal of Information Science Vol 36, Issue 3, pp. 306 - 323

[18] Socher, Richard, Chen, Danqi, Manning, Christopher D. and Ng, Andrew Y. Reasoning with neural tensor networks for knowledge base completion. In Advances in Neural Information Processing Systems, 2013a.

[19] J. Boyd-Graber, H. Daumé & M. Iyyer, “Generating Sentences from Semantic Vector Space Representations”, 2014

[20] J. Martens, “Deep learning via Hessian-free optimization”, In Proceedings of the 27th International Conference on Machine Learning (ICML). ICML 2010, 2010.

[21] I. Sutskever, J. Martens and G. Hinton, “Generating Text with Recurrent Neural Networks”, 2011

[22] "The Unreasonable Effectiveness of Recurrent Neural Networks", Karpathy.github.io, 2017. [Online]. Available: http://karpathy.github.io/2015/05/21/rnn-effectiveness/.

[23] "Recurrent Neural Networks Tutorial, Part 1 – Introduction to RNNs", WildML, 2017. [Online]. Available: http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/.

[24] "An overview of word embeddings and their connection to distributional semantic models - AYLIEN", AYLIEN. [Online]. Available: http://blog.aylien.com/overview-word-embeddings-history-word2vec-cbow-glove/.

[25] Corrado, Greg, and Jeffrey Dean. "Distributed Representations Of Words And Phrases And Their Compositionality". N.p., 2017.

[26] Y. Zhang, R. Jin and Z. Zhou, "Understanding Bag-of-Words Model: A Statistical Framework". [Online]. Available: https://ai2-s2-pdfs.s3.amazonaws.com/4eb6/00aa4071b9a73da49e5374d6e22ca46eaba6.pdf.

[27] A. Colyer, "The amazing power of word vectors", *the morning paper*, 2016. [Online]. Available: https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/.

[28] J. Collis, "What-is-one-hot-encoding-and-when-is-it-used-in-data-science", *https://www.quora.com*. [Online]. Available: https://www.quora.com/What-is-one-hot-encoding-and-when-is-it-used-in-data-science.

[29] A. Colyer, "GloVe: Global Vectors for Word Representation", *the morning paper*. [Online]. Available: https://blog.acolyer.org/2016/04/22/glove-global-vectors-for-word-representation/.

[30] S. Gouws, "How-is-GloVe-different-from-word2vec", *https://www.quora.com*. [Online]. Available: https://www.quora.com/What-is-word-embedding-in-deep-learning.

[31]"An overview of word embeddings and their connection to distributional semantic models - AYLIEN", *AYLIEN*. [Online]. Available: http://blog.aylien.com/overview-word-embeddings-history-word2vec-cbow-glove/.

[32] "Google Code Archive - Long-term storage for Google Code Project Hosting.", *Code.google.com*. [Online]. Available: https://code.google.com/p/word2vec/.