National College of Ireland

End-to-End dialogue systems with Dynamic Memory Networks and FastText

MSc Research Project
Data Analytics

Sukanya Hanumanthu
x17103886

School of Computing
National College of Ireland

Supervisor: Dr.Pramod Pathak,Dr.Paul Stynes,Dympna O'Sullivan

National College of Ireland
Project Submission Sheet – 2017/2018
School of Computing

Student Name:	Sukanya Hanumanthu
Student ID:	x17103886
Programme:	Data Analytics
Year:	2018
Module:	MSc Research Project
Lecturer:	Dr.Pramod Pathak,Dr.Paul Stynes,Dympna O'Sullivan
Submission Due Date:	13/08/2018
Project Title:	End-to-End dialogue systems with Dynamic Memory Networks and FastText
Word Count:	5291

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.

Signature:	
Date:	16th September 2018

PLEASE READ THE FOLLOWING INSTRUCTIONS:

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. **You must ensure that you retain a HARD COPY of ALL projects**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.
3. Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

End-to-End dialogue systems with Dynamic Memory Networks and FastText

Sukanya Hanumanthu
x17103886
MSc in Data Analytics
National College of Ireland

16th September 2018

Abstract Conversational dialogue systems act as the foremost layer of contact of an AI system while making machine-human interactions. Traditional dialogue systems incorporate a modular approach which demand for handcrafting of each module thus increasing the chances of propagating errors from one module to the other. To overcome this, an end to end dialogue system with Dynamic memory Network(DMN) for Question Answering was constructed. This project thrives to improve the performance of DMN by using FastText word to vector conversion technique and compares it with the DMN implemented using global vectors. Results obtained after testing on five different types of Question formats prove that FastText performs well with at least 4% increase in the accuracy. If the same technique was implemented for all dialogue systems prior to building a model, then a gradual shift in the improvement conversational systems can be observed.

1 Introduction

The process of enabling machines to converse with humans in a natural way is one of the key problems in today's Artificial Intelligent systems. In the past few years, models were built based on the features designed by hand engineering given by a set of business rules, thus are domain specific and stereotypical. Advancements in deep learning have added a new twist allowing the models to be built directly from a massive amount of conversational data capturing hidden feature interactions though they have not reached perfection (Goodfellow et al. (2016)).

1.1 Background and Motivation

Dialogue systems will become indispensable with their wide variety of use cases including recommender systems integrated with text to voice conversion, computer vision, information retrieval, robotics according to Milhorat et al. (2017). Non-task-oriented models can also be used for features that does not consist a measurable goal for example, entertaining, language learning or for gaming purposes(Young and Williams (2013)). Existing systems are still in their infancy which are far from passing the Turing test of not identifying whether the conversation is handled by a machine or human (Ram et al. (2018)).

Even the question answering (QA) aspect require deeper Language Understanding (LU) involved with critical Natural Language Processing (NLP) providing a response generated over relevant facts gathered (Kumar et al. (2016)).

The motivation for moving towards an end to end building approach is that the state of the art modular approach demand a lot of hand-crafting, time and cost. The traditional approach has various components including Automatic speech recognizer, Natural language interpreter, Dialogue state tracker, Dialogue response selection, Language generator and a text to speech converter as shown in the figure 1.

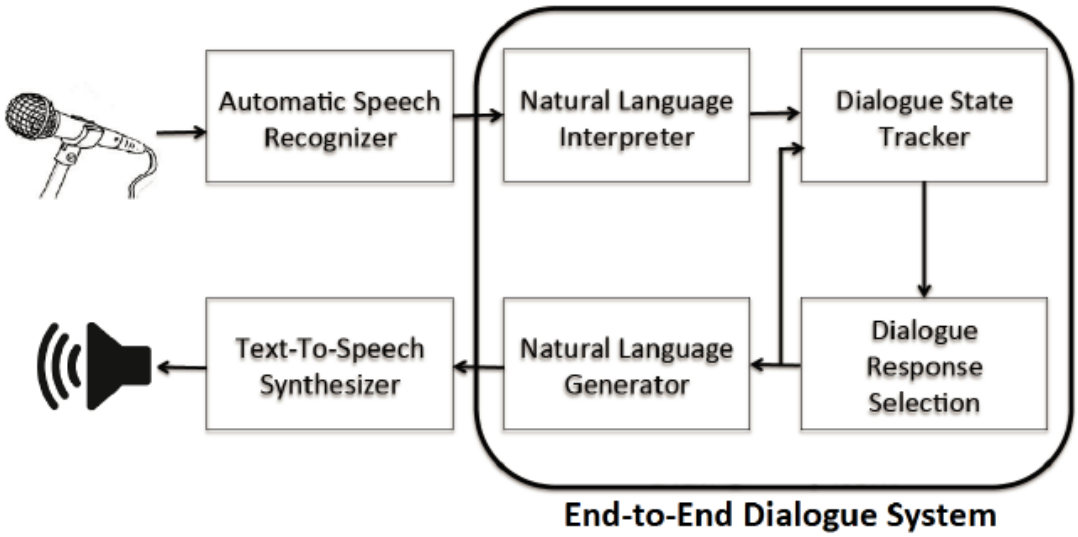

Figure 1: Traditional vs End-to-End dialouge systems
(Lowe et al. (2017))

Each of the mentioned modules have their specific functions trained individually and are evaluated accordingly, for example a state tracker is used for predicting slot filling pairs which measured using cross entropy whereas a response generator is measured using log-likelihood based on the relevancy of output response(Serban, Lowe, Henderson, Charlin and Pineau (2015)). Also, the modular approach tends to propagate errors from one level to the other and the system is not robust to the overall error scenario(Li, Chen, Li and Gao (2017), Zhang et al. (2018)). On the other hand, an end to end approach has been used which directly takes the input generating a response instead of passing through all the modules. The main motivation is to improve a Dynamic Memory Networks Question Answering model by changing its existing word to vector strategy that communicates naturally understanding the scenario, so that it can be integrated with any task-oriented systems (Ritter et al. (2011)).

1.2 Research Formulation

Dialogue system is a computer which can handle meaningful conversations with humans and they are categorized into two types including task-oriented and non-task-oriented

dialogue systems. Task-oriented systems are built to fulfil a pre-defined set of tasks, examples include personal assistants like Microsoft Cortana, Apple Siri, Amazon Alexa, Google assistant etc., (Ram et al. (2018)) for finding products, restaurant table booking or for general search. A Dynamic Memory Network (DMN) is an end to end Question Answering (Q/A) system for Natural Language Processing (NLP).

The model was built on Facebook dataset having human-human conversations collected in text format which is further sent to word to vector models for extracting the relationship between the words in the vector space. This paper compares the model performance by using fastText and global vector (gloVe) word to vector conversions that support non-task-oriented systems using memory networks which can hop back to remember a conversation. This paper stresses on Question Answering (QA) aspect of non-task-oriented systems using DMN.

The aim of this research is to investigate if fastText word to vector conversion technique on an end to end DMN approach can improve the performance compared to global vector technique.

This research contributes to Question Answering aspect of non-task-oriented systems. This data driven approach cannot rely on the traditional modular way of building dialogue systems using business rules. A significant research can be seen in the field of task-oriented systems towards building personal assistant models using neural networks, long short-term memory, encoder-decoders built using recurrent networks. But the research towards non-task-oriented systems is comparatively less as they are hard to model with no specified goal to achieve and of not having a defined evaluation system which is still an open question and further be discussed in the evaluation section(Ritter et al. (2011)).

1.3 Structure of the paper

The structure of the paper includes the next section with a research on related work focused on the machine learning techniques that have been implemented in building dialogue systems. The later section explains the core methodology of the problem domain and on how the data was extracted, cleaned, pre-processed before sending it to the model. The fourth section explains the implementation part including the explanation of important functions and packages followed by the evaluation of the model. Later, a discussion on the results obtained will analyze the overall scenario of this project and observe key strengths and pitfalls. The final section provides a conclusion and future work aspects.

2 Related Work

This research stands on different pillars including methodologies depending on deep learning, end to end architecture, attention-memory, question answering. Fortunately, there is enough research that has been done in each of these individual aspects which will be further discussed in this section.

Deep learning: Deep learning recently gained attention in the field of dialogue systems with the increased hardware support which made data-driven neural network models to be built. Also, the data required to support these models was made available through a survey on available corpora which was initiated to gather dialogue corpus from various

researchers (Serban, Lowe, Henderson, Charlin and Pineau (2015)). Various deep learning techniques were applied in designing different components of dialogue systems. For example, Recurrent Neural Network (RNN) has been used for intent classification and to fill slots to label the domain(Liu and Lane (2016)). Slot filling is a challenging task pertaining to language understanding which aims to fill certain labels to derive and achieve a task(Deng et al. (2012)). Slot ID and deep belief networks (DBN) have performed well in this compared to RNN (Deoras and Sarikaya (2013)). While RNN was used for sentiment analysis (Socher et al. (n.d.)), response generation (Serban et al. (2016)), parsing (Socher et al. (2015)) 2011, inferencing (Bowman et al. (2015)) and for implementing attention mechanism (Xing et al. (2017)). But the above models do not have any memory modules and were not able to deal with question answering as all the tasks cannot be solved with one neural network even though RNN was arranged in a Hierarchical Encoder-Decoder (HRED) (Serban, Sordoni, Bengio, Courville and Pineau (2015)).

RNN was proved to be efficient in speech recognition (Hinton et al. (2012)), image recognition (Lecun et al. (1998)) and context modeling (Mikolov and Zweig (2012)). Huwaei technologies have introduced a neural responding machine built using RNN with latent representation for short text and context classification which showed 75% accuracy in generating responses with no grammatical errors (Shang et al. (2015)). RNN has been used in sequence to sequence models focused on machine translation where one-layer acts as an encoder which takes the input in one language and sends it to the other layer for converting it to another language(Kalchbrenner and Blunsom (2013)). The approach used is like the one used for language modeling which was implemented to overcome the problem of vector conversions at every level of deep neural network (Sutskever et al. (2014)). These models combine Long Short-Term Memory (LSTM) mechanism with RNN which seems like DMN approach except that they do not have episodic memory module to store recently used dialogues but they tend to generate variant answers handling the issue of duplicate answer generation(Wen et al. (2015), Mikolov et al. (2010)).

End-to-End: The aim to process interdependence between the individual components in a traditional dialogue system architecture is hard to achieve which demand significant human effort. On the other hand, these modular systems are not robust to errors, thus transmitting them to consecutive modules degrading the overall system performance. The above-mentioned points made the researchers to switch to an end to end paradigm where a complete model is trained from inputs till the output using a huge corpus instead of building one module after the other(Zhao and Eskenazi (2016)). The end to end models mostly rely on deep learning techniques which can handle huge data. An end to end encoder decoder was implemented on the ubuntu dialogue corpus having binary conversations trained based on a single objective function. This method needs a lot of pre-processing before feeding the model with the corpus(Lowe et al. (2017)). Other end to end systems include negotiation dialogue management by decoding the inputs and training further instead of depending on just the likelihood which showed better results. This approach is different in the aspect where the language agents work individually on negotiations making the model to learn reasoning and language skills in an unsupervised manner(Lecun et al. (1998)).

Another approach which did well with language understanding engulfs both Reinforcement Learning (RL) and RNN forming a Deep Recurrent Q Network (DRQN) belonging to task-oriented systems to amplify the learning speed in gaming applications trained on supervised data. This model outperformed the state-of-the art approach and replaced

three modules including Dialogue State Tracking (DST), Dialogue Policy and Natural Language Understanding (NLU)(Zhao and Eskenazi (2016)). But the DRQN cannot be generalized to other domains as it was trained to handle only short binary questions. To overcome this, a new model was built by Microsoft researchers using a neural dialogue mechanism with a user simulator to understand the goal using slot filling by NLU and Dialogue Management (DM) with the use of reward-penalty approach of RL (Li, Chen, Li, Gao and Celikyilmaz (2017)). Most of the RL based models focus on dialogue policy learning and are used in task-oriented applications such as online shopping integrated with recommender systems (Yan et al. (2015)) but none of these models hold any mechanism to use the past and recent scenarios to answer the upcoming questions.

Attention and Memory: The backbone for DMN is memory networks with its function of storing the input words or sentences. This section compares other architectures which hold memory and attention mechanisms to analyse similar models further. Attention mechanism was first applied in translation (Bahdanau et al. (2014)) and the same approach was applied to response generation in single turn (Shang et al. (2015),Vinyals and Le (2015)). A hierarchical recurrent attention network (HRAN) was recently proposed to extract important vectors within the utterances. This model does well with attention on both word and utterance level having one level for word to vector conversion over utterances getting the context of conversation and the resultant being sent to attention level of utterance and context to decode the response(Xing et al. (2017)). Another hierarchical attention model was developed by(Yang et al. (2016), Yan et al. (2015)) is designed for document classification, a similar model was made for image classification (Stollenga et al. (2014)), phrase representation (Cho et al. (2014)), caption tagging (Li, Chen, Li and Gao (2017)) and object identification(Seo et al. (2016)), These models were implemented for response generation under specified perspectives like classification, diversity and response generation and were not integrated with memory modules like DMN.

Memory mechanism was implemented in Turing memory machines to solve algorithmic expressions and recently proposed memory networks have added a memory component to deal with Question Answering(Weston et al. (2014),Weston et al. (2015)). This memory component is divided to individual parts handling input, response, generalization and mapping. But this model must be trained with sentences individually but not in a sequence manner and need to use n-gram feature vector along with mechanism to align sentences in an order. Whereas DMN has a sequential processing that captures the position of a sentence so that it can be applied to a broader range of applications without any further processing.

Question-Answering and word to vector conversions: Popular approaches to implement this model include the use of a connected database and slot filling, Knowledge base with online access to search engines, dependency trees or just neural networks and sentences. Several web-based Question Answering engines were developed on open and closed domain specific ontology(Kakar and Kandpal (2013),Kwok et al. (2000),Katz et al. (2002),Kwok et al. (2000),Etzioni et al. (2008)) another set includes question-based models which are designed with specific strategies to locate the answers(Hovy et al. (2000)). A mix of neural logic reasoning was implemented over a knowledge base for visual answering based on images (Andreas et al. (2016)). These QA systems are hard to debug when a right answer is not produced as they do not have a repository of facts. The above QA models have not made emphasis on the word to vector conversion part of the model.

Most of these models use word embeddings which are formed using a skip-gram model consisting of updates held in the word embedding matrix. These word vectors present in the embedding matrix have semantic and syntactic details that are captured (Mikolov, Le and Sutskever (2013),Bengio et al. (2017),Mikolov, Chen, Corrado and Dean (2013)). A process called WECOSim was introduced to transform a question to a bag of embedded words in which the word representation is previously analysed using bag of words and a cosine similarity between the questions will be taken into consideration to compare with the existing questions set(Othman et al. (2017)). This approach works well with a repository of less questions and cannot be generalized where as it can be done using DMN along with fastText word to vector conversion.

3 Methodology

This section describes the overall approach followed to build and evaluate an End-to-End DMN model. Deep learning is used for designing the model by following knowledge data discovery in databases (KDD) to convert data to knowledge (Fayyad (1996)). The process of KDD is as shown in the figure 2. It includes an explanation of data extraction, processing, transformation and interpretation techniques which are detailed further. Machine learning can be performed in two ways, one with labelled data (Supervised) and other with no labels (Unsupervised). Supervised data has been used to train the model in this project.

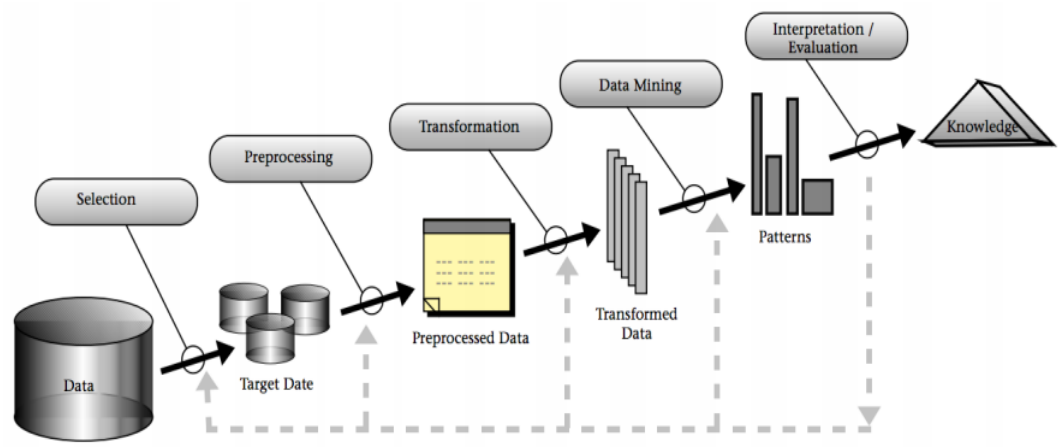

Figure 2: KDD overview
(Fayyad (1996))

3.1 Data Selection and Extraction

The first step of KDD is to select and extract data from the source for further processing. The corpus to build conversational dialogue systems are scarce until they were made available to the public in 2015 by Serban's team (Serban, Lowe, Henderson, Charlin and Pineau (2015)) encouraging the current progress in dialogue systems. Large datasets are required to build a perfect dialogue system, such corpus from various researchers were collected and open-sourced on Git after a survey on available data from which the bAbI tasks dataset was selected for QA model. This data set is released by Facebook AI

research which has 20 different subsets each having 1000 dialogues to test the language understanding of a model(Othman et al. (2017)). This dataset is ideal as it was proved to be significant in several experiments to test the model's logical reasoning for building an end to end systems (Bordes et al. (2016),Madotto et al. (2018)).

3.2 Pre-processing

The dataset extracted has conversations along with questions, answers and facts. The data was randomly arranged, and it was sent for a sanity check to make sure that there are no racist comments or unwanted hate speech. The sanity check was done by sending each of the train, test and fact sets data to a python code which tests for a list of words to be removed. This was done by parsing through every line to check for the bad words given. This filtering mechanism again has two functions, one for enabling the word to be deleted entirely or with a second function to replace the bad word with any other word or blank. Fortunately, there were not many bad words or hate speech in this dataset but still followed this procedure to make it perfect.

3.3 Transformation

At this stage, the data is completely clean which is ready to get transformed to vectors. The model is highly dependent on the vector representations that are obtained and trained. These vectors are used to find the Euclidean distance or cosine similarity which shows meaningful relationships between sentences. The base model will be constructed using glove pre-trained vectors which was obtained by scraping Wiki pages to draw word to vector representations. The cleaned data was first trained using the glove data to form vectors within. The same procedure was again performed using fastText which is a novel method in any QA model to get vector representations. FastText is popular for word classification, topic modelling and word representations which was developed by Facebook research [28]. FastText is considered more efficient than glove because of the following reasons:

1. FastText is better at finding vector representations (word embeddings) even for rare words.

2. Efficient in producing vector representations that are not in the input dataset by breaking the word to character n-gram which glove cannot do (Joulin et al. (2016),Bojanowski et al. (2017)).

3. The FastText calculates character n-grams which are then taken average for a complete word. This approach with character n-gram performs better than word2vec and glove(Bojanowski et al. (2017)).

Based on the above reasons, FastText was implemented for vector conversions. A pre-trained 6GB of FastText data obtained from scraping through Wiki news and articles was collected to perform conversions for the bAbI dataset.

3.4 Data Mining

DMN model was constructed for the transformed data in this step. DMN is previously implemented for question answering (QA) using glove vectors in 2015 (Kumar et al.

(2016)) which was later modified and used for object identification in images for QA purposes (Xiong et al. (2016)). DMN takes the vectors as inputs to generate a proper response. It is a neural network-based approach which is even applicable for text/image classification. The DMN was later modified in the year 2016 by adding input fusion layers for image analysis which showed 60.3 percent accuracy for image QA. This section gives an overview about the DMN modules with their functionality to understand the implementation part explained in the next section.

DMN has fours modules as shown in the figure 3 which include input, answer, question and episodic memory. Each of these modules are explained as below:

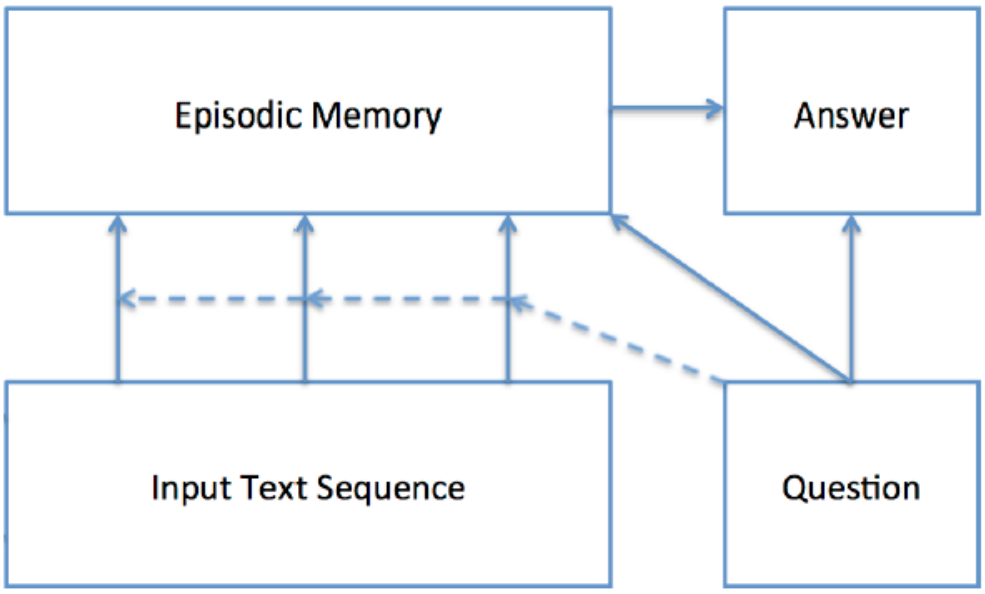

Figure 3: DMN overview
(Kumar et al. (2016))

Input Module: The project is based on natural language processing, so the input could be any form of text either an article, book, dialogue set or a review. The basis is all dependent on the vector representation of the text. A recurrent neural network (RNN) takes the inputs as word embeddings that are generated after the transformation stage in which each sentence consists a set of word vectors.

1. At every step of given input, the hidden state of the network gets updated with the new embedding matrix.

2. In case, if the input is just one sentence, then the output equals the hidden states of the network.

3. If a sequence of sentences is given as input, then the hidden state is updated with all the word tokens sequentially.

A gated recurrent network (GRU) has been used as LSTM is computationally expensive, time taking, prone to overfitting (Xiong et al. (2016),Hochreiter and Schmidhuber (1997)).

Question module: This is similar to input module which take vectors for further processing. A given question is first encoded to vectors but in this project, a test set was created which has vector representations and was directly given to question module. The question module also has a GRU which updates on the hidden state for every new question producing the embedding matrix for every word. The embedding matrix is shared between question and input modules with the only difference that the output of question module is equivalent to the last hidden state of GRU.

Episodic memory module: This module iterates over the input vectors which are needed for the questions based on the attention mechanism.

1. It observes the previous questions and stores the vectors that are important in the episodic memory.
2. Memory updates are managed by recurrent network and attention mechanism. Each episode ei is formed by considering attention mechanism over facts c, questions q and memory of the previous iteration mi-1.
3. Episodic memory is updated in GRU for every iteration. Initial memory state of GRU is assigned to the first question q.
4. The final episodic memory serves as the answer after certain iterations over the input vectors.

Due to these multiple iterations, various input vectors will be visited during each pass allowing to extract indirect inference.

Attention and memory update mechanism: The attention mechanism is implemented using a gate function that is derived from fact, given question and previous memory with a scoring function. A scoring function takes feature vector set as an input that captures similarities between input, question and memory vectors. Memory updates depend on input vector and gates function having the episodic memory vector given to the answer module in the final state of GRU as explained in the DMN. After building the End to End DMN as discussed above, the input vectors will be sent to get the relevant output.

This deep learning approach takes significant processing time and data to make the model perform well in various scenarios using Google cloud, tensorflow but in this case, the project was completely run on the CPU.

3.5 Interpretation

The constructed model is tested against various scenarios and types of questions using the bAbI dataset. The data was divided into training and testing with which it was tested through random shuffling of the data up to 10 iterations with the accuracy captured. Further discussion regarding the evaluation and results has been given in the next sections.

4 Design and Implementation

This section shows the procedure followed to make the DMN model with FastText and make it run. The project mostly used python and a little of bash while extracting the data. For this project, three CPU's were used in which PyCharm, Python 2.7 interpreter and a bash console were used.

4.1 Extraction

The data is extracted using a simple bash command to hit a URL to download the bAbI tasks dataset consisting of 20 types of question files each of which having their own testing, training and fact data.

4.2 Sanity filter

After downloading the data, a sanity filter was made in Python 2.7 version by using re and random packages. This sanity filter accepts a list of words that needs to be filtered. The call for the sanity class has been made after taking the text file as input and parsing each line by making a call to the clean function. There are two functions in the sanity class including cleaning and replace which were built to replace bad words with another word or to just remove the word without replacement upon the user selection. A regular expression was used to find the occurrence of the word in the text in any form.

4.3 Vector conversion

To perform this task, few packages named Theano, NumPy, Lasagne and Pickle must be downloaded. Tensor package should be imported from theano before starting with the vector conversions. The cleaned text was sent to be trained with the use of glove and fastText. It starts with downloading the pre-trained wiki data vector set for both. Then a code has been written to split each line in the data to capture words and their corresponding vectors which are further added to a function which makes a call for the bAbI dataset. The inputs from the bAbI tasks are then converted to vectors based on the vector set given. The main vector conversion calls for other functions which can deal with missing words in glove and fastText. The vectors were extracted using the tensor function of theano by getting normalized or constant vectors by passing the dimension, word vector size. The same procedure is followed for all the modules including episodic memory, input, question and answer.

4.4 DMN construction

Constructing DMN: DMN was constructed using all the components described earlier using the vectors and dot product between them forming the correct formula with softmax activation function to minimize the cross entropy. The vectors are then fed to the DMN model for final testing. The input is given using the training, testing and fact dataset. The data is shuffled continuously after every epoch and was trained again.

5 Evaluation

The Facebook bAbI dataset has been always tested using their testing dataset provided. There are no fixed evaluation methods yet designed for dialogue systems which is a huge problem and is stillan open question. Other techniques like Bilingual evaluation score Galley et al. (2015), next utterance classification(Landeghem (2016)) have been used in different cases but the results do not match human evaluation. So, based on the previous experiments on QA, accuracy obtained from training and testing is considered(Lowe et al. (2017)). In this project, five different questioning scenarios were taken in which

the training, testing and facts data were collected and converted to vectors. Two models were built with glove and FastText which were tested against the corresponding testing dataset. Only 10% of training data was used to train the model. The processing times were also captured along with the accuracy and confusion matrix. The base model is proposed in DMN having glove with which the results will be compared with fastText DMN. Below figure shows the output confusion matrix and accuracy for the bAbI task 1 with glove and fastText.

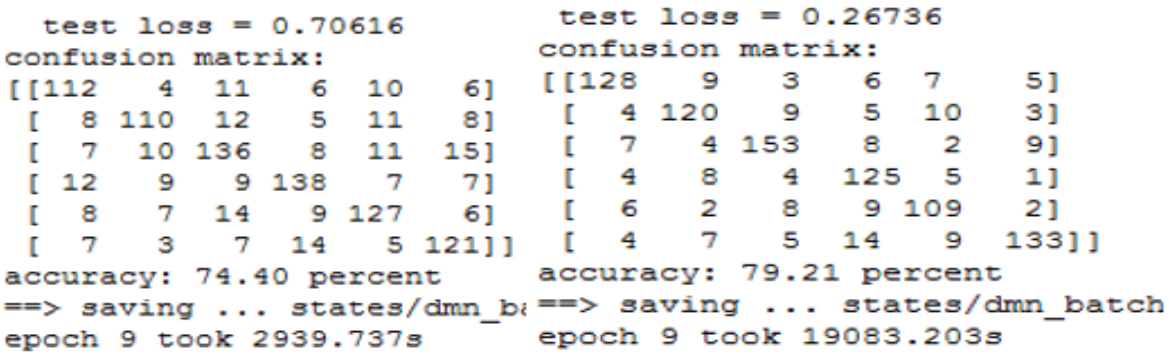

Figure 4: Confusion matrix and accuracy of DMN for one supporting fact data with glove on the left and FastText on the right side

The above figure for task1 with only one fact given after testing and training the data. The accuracy of the model has increased by over 5% with fastText compared to the base model. The time taken for training and testing in one iteration is 2939.737 seconds for glove which is equal to 49 minutes and there are 8 more iterations before the final accuracy has computed. The processing time for fastText is even more as pre-trained fastText vectors are much larger in this case.

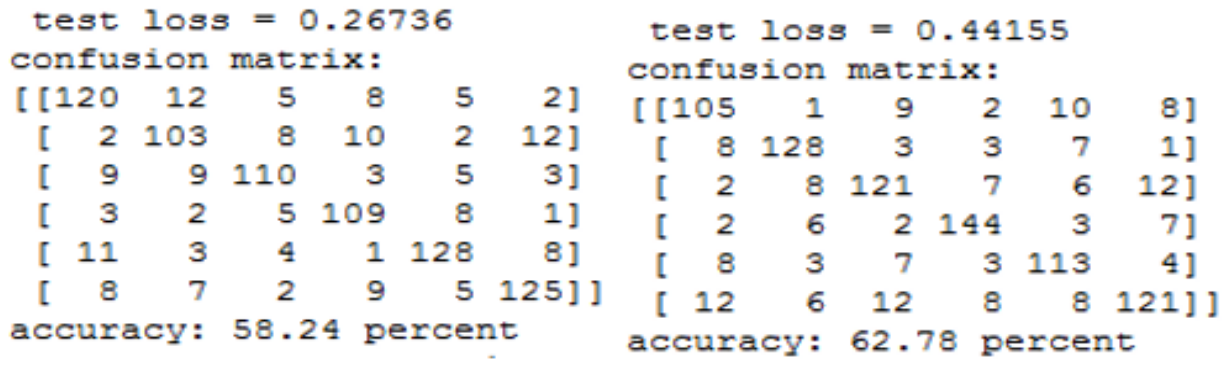

Figure 5: Confusion matrix and accuracy of DMN for two supporting fact data with glove on the left and FastText on the right side

A similar experiment was done by using two supporting facts which showed that the DMN's accuracy is 58.24% with glove and has just increased to 62.78% with fastText. In figure 6 shows the result from testing task 3 with three supporting facts was able to easily iterate over input and facts to give a decent number of correct answers. The accuracy

with fastText is more in this scenario by 4.1% but it was not performing as well as with the one supporting fact which is unexpected.

```
confusion matrix:
[[132   4   4   5   1   6]
 [  9 110   6   2   6   7]
 [  7   8 135   5   2   1]
 [  1  12   1 112   5  11]
 [  6   3  11  15 147   3]
 [  3   9   7   2   1 128]]
accuracy: 62.22 percent
```

```
confusion matrix:
[[113  14   3   5   6   2]
 [ 10 107   8   4   4   7]
 [  7   1 125  13   5   5]
 [  9  19   5 117   3  10]
 [  4   3   6  15 122   6]
 [  2   5   7   1   6 134]]
accuracy: 66.32 percent
```

Figure 6: Confusion matrix and accuracy of DMN for three supporting fact data with glove on the left and FastText on the right side

```
confusion matrix:
[[116   7   9  11   5   6]
 [ 12 136   5   8   4  10]
 [  5  14 127   8  14   9]
 [  9   5   8 134   6   4]
 [ 13   5  11   1 121   4]
 [  4   9  14   7   7 132]]
accuracy: 76.60 percent
```

```
confusion matrix:
[[124  3   3  15   5   3]
 [  9 111  12   5   5  11]
 [  3   2 102   2  12   2]
 [ 12  9   8 132   3   7]
 [  9   7   3   6 108   5]
 [ 11  4   5   3   9 101]]
accuracy: 82.22 percent
```

Figure 7: Confusion matrix and accuracy of DMN for two argument relations data with glove on the left and FastText on the right side

In figure 7, it shows that the accuracy is 82.22% for DMN and fastText for bAbI task4 having two arguments which gives a relation to the question asked in the conversation. This performed 6% better than the DMN glove.

```
confusion matrix:
[[124   3   7  10   2   8]
 [  3 106   5   3   7  11]
 [  9   5 110   7  12   1]
 [ 13   2   6 101   6   3]
 [  9  15   5   6 124   6]
 [  8   9  11   7   1 152]]
accuracy: 77.33 percent
```

```
confusion matrix:
[[105   7   9   7   8   7]
 [ 11 150   6   8   2  10]
 [  1  18 128   7   9   8]
 [  8   1   1 114   8   4]
 [  2   6   6   8 104   9]
 [  2   4  17   5  12 118]]
accuracy: 85.21 percent
```

Figure 8: Confusion matrix and accuracy of DMN for three argument relations data with glove on the left and FastText on the right side

Figure 8 below depicts the accuracy obtained for the last task handling three argument relations with glove which is slightly more than its previous accuracy attained with two arguments equaling to 77.33% and with the fastText it has reached to 85.21%.

Overall view of results: Below is the overall result summarized after testing with various question types.

Question type	DMN glove	DMN FastText
one supporting fact	74.40%	79.21%
two supporting facts	58.24%	62.78%
three supporting facts	62.2%	66.32%
two argument relations	76.60%	82.2%
three argument relations	77.33%	85.21%

Table 1: Results of DMN with glove and FastText

It is evident that there is an improvement of around 4-6% with the use of FastText but this would be more accurate when tested with huge corpus. The overall accuracy ranged in the case of DMN with fastText ranged from 62 to 85 in five of the cases. It performed extremely well with three argument relations but more analysis has to be done to improve the accuracy for two supporting facts.

6 Discussion

The results obtained show that there is at least 3-6% increase with fastText in the accuracy over five different question types including single supporting fact, two supporting facts, three supporting facts and with two and three argument relations tasks. As expected, there is certainly a raise compared to glove but this could be made even better with the increase in data that is being tested and trained. An interesting fact is that, with the increase in the number of facts provided, the performance of DMN model has raised.

The dataset has ten thousand dialogues per task each for test, train and facts which does not have many rare words and are not in the overall vector space. Whereas if the approach tested on an extremely large corpus having at least ten hundred thousand dialogues with rare words like biological and medical terms which were not used frequently, then DMN with fastText will serve its best. Therefore, the null hypothesis which states that the fastText cannot improve the DMN QA performance can be rejected in this research. This proved to be better than the results obtained with glove (DMN).

The use of fastText has improved the performance of DMN but the processing time is extremely slow taking over 7.35 hours for glove per task and is double in case of fastText. So, the overall time spent just for testing 5 different tasks in glove and fastText took days on a CPU. This problem can be avoided by setting up TensorFlow on a GPU or setting up a cloud environment on Google Cloud or AWS. These two approaches were tried and were not easy to implement as the existing GPU type did not match the TensorFlow configuration and the cloud setup consumes time.

7 Conclusion and Future work

FastText has been mainly used for text classification and is not used in any of the dialogue systems for word to vector conversions. The end to end models are sensitive and should be carefully built considering every aspect including the word embeddings. It was clearly mentioned in DMN that the model is highly dependent on the input vectors, if that is the scenario then it is always efficient to use the best possible methods for it which was done in this research. The key findings show that there is a definite increase in the accuracy by at least 4% tested on five different question answering scenarios. This work contributes to the improvement of dialogue systems by replacing the vector conversion techniques and to thrive to find any other vector conversion methodologies which will further improve this domain.

A web user interface with DMN and fastText on the backend can be built to test on real human users asking for a rating on different aspect of the system including the grammar, relevancy, context maintenance, uniqueness of the answers etc. This will further give a real experience on the system performance raising various questions to make the model even more powerful. Secondly, a large dataset can be used to train the system even on rare words which will certainly perform well with fastText. Thirdly, the same model can be built using big data on distributed systems, cloud environment and high-performance computing systems for faster processing and for making an efficient conversational system. Lastly, there is no data available with facts, questions and answers that is much larger to test the model. So, data in large volumes should be gathered to make it available for this kind of research.

8 Acknowledgements

I would like to thank my supervisor Dr.Paul Stynes for steering me in the right direction by giving interesting strategy classes on thesis writing and implementation which acted as a backbone for my project. I thank Dympna O'Sullivan for giving her valuable time and feedback in reviewing the document, code and results every week with patience. I thank Dr.Pramod Pathak for his motivational support throughout this project. I must express my profound gratitude towards my parents and my college friends for continuous encouragement throughout my research.

References

Andreas, J., Rohrbach, M., Darrell, T. and Klein, D. (2016). Learning to Compose Neural Networks for Question Answering, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, San Diego, California, pp. 1545–1554.
URL: *http://www.aclweb.org/anthology/N16-1181*

Bahdanau, D., Cho, K. and Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate, *arXiv:1409.0473 [cs, stat]* . arXiv: 1409.0473.
URL: *http://arxiv.org/abs/1409.0473*

Bengio, Y., Ducharme, R., Vincent, P. and Jauvin, C. (2017). A Neural Probabilistic Language Model, p. 19.

Bojanowski, P., Grave, E., Joulin, A. and Mikolov, T. (2017). Enriching Word Vectors with Subword Information, *Transactions of the Association for Computational Linguistics* **5**: 135–146.
URL: *https://transacl.org/ojs/index.php/tacl/article/view/999*

Bordes, A., Boureau, Y.-L. and Weston, J. (2016). Learning End-to-End Goal-Oriented Dialog, *arXiv:1605.07683 [cs]* . arXiv: 1605.07683.
URL: *http://arxiv.org/abs/1605.07683*

Bowman, S. R., Potts, C. and Manning, C. D. (2015). Recursive Neural Networks Can Learn Logical Semantics, *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, Association for Computational Linguistics, Beijing, China, pp. 12–21.
URL: *http://www.aclweb.org/anthology/W15-4002*

Cho, K., van Merrienboer, B., Bahdanau, D. and Bengio, Y. (2014). On the Properties of Neural Machine Translation: Encoder-Decoder Approaches, *arXiv:1409.1259 [cs, stat]* . arXiv: 1409.1259.
URL: *http://arxiv.org/abs/1409.1259*

Deng, L., Tur, G., He, X. and Hakkani-Tur, D. (2012). Use of kernel deep convex networks and end-to-end learning for spoken language understanding, IEEE, pp. 210–215.
URL: *http://ieeexplore.ieee.org/document/6424224/*

Deoras, A. and Sarikaya, R. (2013). Deep Belief Network based Semantic Taggers for Spoken Language Understanding, p. 5.

Etzioni, O., Banko, M., Soderland, S. and Weld, D. S. (2008). Open information extraction from the web, *Communications of the ACM* **51**(12): 68.
URL: *http://portal.acm.org/citation.cfm?doid=1409360.1409378*

Fayyad, U. (1996). From Data Mining to Knowledge Discovery in Databases, p. 18.

Galley, M., Brockett, C., Sordoni, A., Ji, Y., Auli, M., Quirk, C., Mitchell, M., Gao, J. and Dolan, B. (2015). BLEU: A Discriminative Metric for Generation Tasks with Intrinsically Diverse Targets, p. 6.

Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep Learning*, MIT Press. `http://www.deeplearningbook.org`.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N. and Kingsbury, B. (2012). Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups, *IEEE Signal Processing Magazine* **29**(6): 82–97.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory, *Neural Comput.* **9**(8): 1735–1780.
URL: *http://dx.doi.org/10.1162/neco.1997.9.8.1735*

Hovy, E., Gerber, L., Hermjakob, U., Junk, M. and Lin, C.-y. (2000). Question Answering in Webclopedia, *In Proceedings of the Ninth Text REtrieval Conference (TREC-9*, pp. 655–664.

Joulin, A., Grave, E., Bojanowski, P. and Mikolov, T. (2016). Bag of Tricks for Efficient Text Classification, *arXiv:1607.01759 [cs]* . arXiv: 1607.01759.
URL: *http://arxiv.org/abs/1607.01759*

Kakar, V. K. and Kandpal, M. (2013). Techniques of Acoustic Feature Extraction for Detection and Classification of Ground Vehicles, **3**(2): 8.

Kalchbrenner, N. and Blunsom, P. (2013). Recurrent Continuous Translation Models, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Seattle, Washington, USA, pp. 1700–1709.
URL: *http://www.aclweb.org/anthology/D13-1176*

Katz, B., Felshin, S., Yuret, D., Ibrahim, A., Lin, J., Marton, G., Jerome McFarland, A. and Temelkuran, B. (2002). Omnibase: Uniform Access to Heterogeneous Data for Question Answering, *in* G. Goos, J. Hartmanis, J. van Leeuwen, B. Andersson, M. Bergholtz and P. Johannesson (eds), *Natural Language Processing and Information Systems*, Vol. 2553, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 230–234.
URL: *http://link.springer.com/10.1007/3-540-36271-1*$_2$3

Kumar, A., Ondruska, P., Iyyer, M., Bradbury, J., Gulrajani, I., Zhong, V., Paulus, R. and Socher, R. (2016). Ask Me Anything:Dynamic Memory Networks for Natural Language Processing, p. 10.

Kwok, C., Etzioni, O. and Weld, D. S. (2000). Scaling Question Answering to the Web, p. 22.

Landeghem, J. V. (2016). Sequence-to-Sequence Learning for End-to-End Dialogue Systems, p. 75.

Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P. (1998). Gradient-based learning applied to document recognition, *Proceedings of the IEEE* **86**(11): 2278–2324.

Li, X., Chen, Y., Li, L. and Gao, J. (2017). End-to-end task-completion neural dialogue systems, *CoRR* **abs/1703.01008**.
URL: *http://arxiv.org/abs/1703.01008*

Li, X., Chen, Y.-N., Li, L., Gao, J. and Celikyilmaz, A. (2017). End-to-End Task-Completion Neural Dialogue Systems, *arXiv:1703.01008 [cs]* . arXiv: 1703.01008.
URL: *http://arxiv.org/abs/1703.01008*

Liu, B. and Lane, I. (2016). Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling, *arXiv:1609.01454 [cs]* . arXiv: 1609.01454.
URL: *http://arxiv.org/abs/1609.01454*

Lowe, R., Pow, N., Serban, I. V., Charlin, L., Liu, C.-W. and Pineau, J. (2017). Training End-to-End Dialogue Systems with the Ubuntu Dialogue Corpus, p. 35.

Madotto, A., Wu, C.-S. and Fung, P. (2018). Mem2seq: Effectively Incorporating Knowledge Bases into End-to-End Task-Oriented Dialog Systems, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Melbourne, Australia, pp. 1468–1478.
URL: *http://www.aclweb.org/anthology/P18-1136*

Mikolov, T., Chen, K., Corrado, G. and Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space, *arXiv:1301.3781 [cs]* . arXiv: 1301.3781.
URL: *http://arxiv.org/abs/1301.3781*

Mikolov, T., Karafiat, M., Burget, L., Cernocky, J. and Khudanpur, S. (2010). Recurrent Neural Network Based Language Model, p. 4.

Mikolov, T., Le, Q. V. and Sutskever, I. (2013). Exploiting Similarities among Languages for Machine Translation, *arXiv:1309.4168 [cs]* . arXiv: 1309.4168.
URL: *http://arxiv.org/abs/1309.4168*

Mikolov, T. and Zweig, G. (2012). Context dependent recurrent neural network language model, *2012 IEEE Spoken Language Technology Workshop (SLT)*, pp. 234–239.

Milhorat, P., Lala, D., Inoue, K., Zhao, T., Ishida, M., Takanashi, K., Nakamura, S. and Kawahara, T. (2017). A conversational dialogue manager for the humanoid robot ERICA, p. 12.

Othman, N., Faiz, R. and Smaili, K. (2017). A Word Embedding based Method for Question Retrieval in Community Question Answering, p. 6.

Ram, A., Gabriel, R., Cheng, M., Wartick, A., Prasad, R., Liu, Q., Nagar, A., Pan, Y., Hwang, G., Khatri, C., Nunn, J., King, E., Song, H., Pettigrue, A., Venkatesh, A., Hedayatnia, B., Bland, K. and Jayadevan, S. (2018). Conversational AI: The Science Behind the Alexa Prize, p. 18.

Ritter, A., Cherry, C. and Dolan, W. B. (2011). Data-Driven Response Generation in Social Media, p. 11.

Seo, P. H., Lin, Z., Cohen, S., Shen, X. and Han, B. (2016). Progressive Attention Networks for Visual Attribute Prediction, *arXiv:1606.02393 [cs]* . arXiv: 1606.02393.
URL: *http://arxiv.org/abs/1606.02393*

Serban, I., Sordoni, A., Bengio, Y., Courville, A. and Pineau, J. (2015). Hierarchical Neural Network Generative Models for Movie Dialogues.

Serban, I. V., Klinger, T., Tesauro, G., Talamadupula, K., Zhou, B., Bengio, Y. and Courville, A. (2016). Multiresolution Recurrent Neural Networks: An Application to Dialogue Response Generation, *arXiv:1606.00776 [cs, stat]* . arXiv: 1606.00776.
URL: *http://arxiv.org/abs/1606.00776*

Serban, I. V., Lowe, R., Henderson, P., Charlin, L. and Pineau, J. (2015). A Survey of Available Corpora for Building Data-Driven Dialogue Systems, p. 56.

Shang, L., Lu, Z. and Li, H. (2015). Neural Responding Machine for Short-Text Conversation, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Association for Computational Linguistics, Beijing, China, pp. 1577–1586.
URL: *http://www.aclweb.org/anthology/P15-1152*

Socher, R., Huang, E. H., Pennin, J., Manning, C. D. and Ng, A. Y. (n.d.). Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection, p. 9.

Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y. and Potts, C. (2015). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank, p. 12.

Stollenga, M., Masci, J., Gomez, F. and Schmidhuber, J. (2014). Deep Networks with Internal Selective Attention through Feedback Connections, *arXiv:1407.3068 [cs]* . arXiv: 1407.3068.
URL: *http://arxiv.org/abs/1407.3068*

Sutskever, I., Vinyals, O. and Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks, *arXiv:1409.3215 [cs]* . arXiv: 1409.3215.
URL: *http://arxiv.org/abs/1409.3215*

Vinyals, O. and Le, Q. (2015). A Neural Conversational Model, *arXiv:1506.05869 [cs]* . arXiv: 1506.05869.
URL: *http://arxiv.org/abs/1506.05869*

Wen, T.-H., Gasic, M., Mrkšić, N., Su, P.-H., Vandyke, D. and Young, S. (2015). Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems, Association for Computational Linguistics, pp. 1711–1721.
URL: *http://aclweb.org/anthology/D15-1199*

Weston, J., Bordes, A., Chopra, S., Rush, A. M., van Merriënboer, B., Joulin, A. and Mikolov, T. (2015). Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks, *arXiv:1502.05698 [cs, stat]* . arXiv: 1502.05698.
URL: *http://arxiv.org/abs/1502.05698*

Weston, J., Chopra, S. and Bordes, A. (2014). Memory Networks, *arXiv:1410.3916 [cs, stat]* . arXiv: 1410.3916.
URL: *http://arxiv.org/abs/1410.3916*

Xing, C., Wu, W., Wu, Y., Zhou, M., Huang, Y. and Ma, W.-Y. (2017). Hierarchical Recurrent Attention Network for Response Generation, *arXiv:1701.07149 [cs]* . arXiv: 1701.07149.
URL: *http://arxiv.org/abs/1701.07149*

Xiong, C., Merity, S. and Socher, R. (2016). Dynamic Memory Networks for Visual and Textual Question Answering, p. 10.

Yan, Z., Duan, N., Chen, P., Zhou, M., Zhou, J. and Li, Z. (2015). Building Task-Oriented Dialogue Systems for Online Shopping, p. 8.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A. and Hovy, E. (2016). Hierarchical Attention Networks for Document Classification, Association for Computational Linguistics, pp. 1480–1489.
URL: *http://aclweb.org/anthology/N16-1174*

Young, S. and Williams, J. D. (2013). POMDP-based Statistical Spoken Dialogue Systems: a Review, *PROC IEEE* **101**(5): 18.

Zhang, S., Dinan, E., Urbanek, J., Szlam, A., Kiela, D. and Weston, J. (2018). PERSONALIZING DIALOGUE AGENTS: I HAVE A DOG, DO YOU HAVE PETS TOO?, p. 14.

Zhao, T. and Eskenazi, M. (2016). Towards End-to-End Learning for Dialog State Tracking and Management using Deep Reinforcement Learning, Association for Computational Linguistics, pp. 1–10.
URL: *http://aclweb.org/anthology/W16-3601*